Transclusion of document fragments from dynamic text

Manu Choudhury

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation
Transclusion of Document Fragments from Dynamic Text

by Manu Choudhury

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science
in the School of Print Media
in the College of Imaging Arts and Sciences
of the Rochester Institute of Technology

May 2010

Primary Thesis Advisor: Professor Patricia Albanese
Secondary Thesis Advisor: Professor Tona Henderson
Permission to Reproduce Thesis

Title of Thesis:
“Transclusion of Document Fragments from Dynamic Text”

I, Manu Choudhury, hereby grant permission to the Wallace Memorial Library of R.I.T to reproduce my thesis in whole or in part, provided the reproduction will not be for commercial use or profit.

Date: May 18, 2010

Researcher’s signature:……………………………………
I would like to thank my primary thesis advisor, Professor Patricia Albanese for her valuable participation, and guidance throughout the research process. I would thank my secondary thesis advisor, Professor Tona Henderson not just for her extremely useful technical advice but also for the mental support that kept me encouraged and confident throughout the process. Both of my advisors have been exceptionally helpful and without them the thesis would not have been possible.

I would also like to thank Steve Battle, who actually gave me the basic idea of this thesis topic. His comments and suggestions were very useful specially when it came to completing the loop in order to finish the actual thesis.

I thank Dr. Scott Williams, my thesis coordinator whose questions and comments ensured that all the efforts converge into a good research thesis rather than a project with only technical goals.

I would also thank Open Publishing Lab, RIT and all the researchers in the lab. We have had very valuable discussions on this topic.

I thank all of my peers for their constant support and debates that I have had with them on the impact of this thesis on the users and thereby making it much more richer.

Last but not the least, I would thank my family, both in India, and in US for their continuing patience, love and support in completion of my thesis.
Table of Contents

List of Figures...........................................................................................................vii
Abstract..................................................................................................................ix
Chapter One Introduction.......................................................................................1
  Statement of the Problem....................................................................................2
  Background and Significance.............................................................................4
  Interest of Study...............................................................................................5
Chapter Two Review of the Literature.................................................................7
  Introduction.......................................................................................................7
  What is Transclusion?......................................................................................8
  Existing Implementations of Transclusion.........................................................10
    Using basic HTML Tags................................................................................10
    Using DOM Nodes.......................................................................................12
    Using Asynchronous JavaScript and XML (AJAX)......................................17
    Using Extensible Markup Language (XML)............................................18
    Using Purple Number – Node Identifier...................................................26
  Ideal Implementation of Transclusion.........................................................27
  Summary.........................................................................................................28
Chapter Three Research Question.....................................................................30
Chapter Four Methodology..................................................................................31
  Design of Algorithm.......................................................................................31
    Overview of the Algorithm..........................................................................31
    Step-by-step Description of the Algorithm.............................................32
    Finding the optimum value of ‘wt’............................................................40
Pseudo Code of the Main Algorithm……………………………………41
Analysis of the Algorithm………………………………………………41
Chapter Five Results…………………………………………………44
Percent Change…………………………………………………………44
Average Accuracy vs. Weights Selected…………………………46
Accuracy of the Final Transcluded Text……………………………52
Accuracy Percentage vs. Change Percentage to the Source Document……54
Accuracy vs. Nature of Change………………………………………55
Chapter Six Summary and Conclusions……………………………………58
Limitations………………………………………………………………………61
Recommendations for further research………………………………62
Chapter Seven Bibliography……………………………………………………63
List of Figures

Figure 1: DOM structure of the above HTML document…………………………13
Figure 2: DOM structure of the source HTML document………………………….15
Figure 3: DOM structure of the source HTML document indicating the selected document fragment…………………………………………………………16
Figure 4: DOM structure of the source XHTML document……………………….18
Figure 5: DOM structure of the user XML document with transclusion tag……………….19
Figure 6: DOM structure of the generated user document (including transcluded fragments from the source document)………………………………………..20
Figure 7: DOM structure of the user XML document using the XPath attribute inside the transclusion tag…………………………………………………..21
Figure 8: DOM structure of the generated user document (corresponding to figure 7)…………………………………………………………………..21
Figure 9: DOM structure of the source document (2\textsuperscript{nd} paragraph changed)……………….23
Figure 10: DOM structure of the generated user document reflecting the change in figure 9……………………………………………………………….23
Figure 11: Transclusion nodes and the document fragment to which they refer………24
Figure 12: DOM structure of the source document (paragraph added between 1 and 2)……………………………………………………………….24
Figure 13: DOM structure of the generated user document reflecting the change corresponding to figure 12……………………………………………….25
Figure 14: Transclusion nodes and the document fragment to which they refer (corresponding to figure 12)……………………………………………………...25
Figure 15: Original source document…………………………………………………...32
Figure 16: Initial selection in the source document………………………………………….32
Figure 17: Changed source document (highlighted portion indicates the new addition)…………………………………………………………………….32
Figure 18: Encryption of the transcluded text corresponding to figure 16 (the numbers in the brackets are the encryption of the outside word)…………………………………..34
Abstract

Transclusion is a concept that allows users to reuse document fragments from different source pages, not by duplicating it, but by including a reference to the original work. Thus, transclusion provides an interesting platform for authors to quote text from other documents in a way such that the quoted text can be compared alongside its original context, and changes in the original document are reflected automatically in the user document. Several researchers have partially implemented transclusion using various techniques and technologies. The issue of transcluding text from a dynamic source page still persists because dynamic content changes have not been reflected.

This research was based on finding a solution to this problem. Encrypted transcluded text, along with its reference, was stored in the user’s document. Using this encrypted message, a document fragment was located in the dynamic source page such that encryption of that fragment matched most closely to the recorded encrypted message. This returned document fragment ensured that changes in the original source documents were reflected in the user’s document. A test set was created with a modified source file, transcluded text, number, nature, and percentage change in the source file. The expected outcome, and the document fragment returned based on the algorithm, were used to determine the accuracy of the algorithm. The average accuracy of the algorithm obtained for the entire test set was found to be 0.92. Moreover, for 60% of the test set the algorithm’s predicted result matched the expected outcome exactly. The algorithm turned
out to be quite forgiving, as even when the source document changed 100%, the average accuracy still turned out to be more than 75%.

Thus, the algorithm can be used as a framework to transclude document fragments from dynamic source pages. The connection between the user document and the source document is retained, and more importantly the changes in the source document are accurately reflected in the user document, thereby allowing users to present their ideas effectively, such that readers can comprehend the ideas presented in the document, and re-use them as a framework for establishing their own ideas.
Chapter One

Introduction

Transclusion is a form of smart inclusion that allows an author to include work of others, not by duplicating the work, but by including a reference to the original work (Nelson, 1995). This concept would not only allow authors to quote document fragments, but also provide a convenient method for the readers to go back to the source page and check the original context themselves. They could compare the quoted text and original context side-by-side as in Ted Nelson's original vision.

Theodor Holm Nelson, known for introducing terms like hypertext, hypermedia, and transclusion, believes that a document is a package of ideas created by human minds. These ideas manifest as text, connections, and diagrams; and so the real objective of a document should be ‘expression, reception, and re-use’ of ideas. According to Nelson, currently the document is just an imitation of paper that cannot be ‘annotated, easily connected, or deeply re-used’ (Nelson, 2007).

For example, in Hyper Text Mark Up Language (HTML), tags like <img>, <object>, <embed>, <iframe> allow users to include remote resources like images, applets, animations, and HTML source pages by means of reference. The problem with these is that while an entire document or object can be referenced, a document fragment may not.
Moreover, there is no specification yet to include text by reference (J, Kolbitsch, & H. Maurer, 2006). The easiest way to refer to remote documents is through hyperlinks. Using hyperlinks one cannot include the source, but can only make a reference to it. Hyperlinks cannot resolve the problem of referencing document fragments either.

Statement of the Problem

Using existing technologies like HTML, Universal Resource Locater (URL), Document-Object-Model (DOM), etc., many researchers have tried to implement transclusion. In all the following implementations, the reference to the selected text is static. Thus, if the source is a dynamic page, the referenced text might not be what the user had initially selected, making the ‘transcluded’ text less practical for the user.

For example: If a user wants to quote a few lines from a paragraph of a document, the static reference would contain the URL of the source document, starting index ($i^{th}$ paragraph, $j^{th}$ line), and offset (number of lines). If a few lines were added to the source page before the quoted text, the static reference would not refer to the text that the user had originally intended.

For example, the source document has the following content:

“The Norwegian Nobel Committee has decided that the Nobel Peace Prize for 2009 is to be awarded to President Barack Obama for his extraordinary efforts to strengthen international diplomacy and cooperation between peoples.

The Committee has attached special importance to Obama’s vision of and work for a world without nuclear weapons.”
Now, if the user wants to quote the first line of the first paragraph, the static reference would be:


Start Index: 1st paragraph, 1st line

Offset: 1 line

Now if the original author adds another line to the source document, and the document now reads as follows:

“Obama has, as President created a new climate in international politics. The Norwegian Nobel Committee has decided that the Nobel Peace Prize for 2009 is to be awarded to President Barack Obama for his extraordinary efforts to strengthen international diplomacy and cooperation between peoples.

The Committee has attached special importance to Obama's vision of and work for a world without nuclear weapons.”

With the reference being still the same, the quote that would be transcluded would read:

“Ohama has as President created a new climate in international politics.”

Instead of:

“The Norwegian Nobel Committee has decided that the Nobel Peace Prize for 2009 is to be awarded to President Barack Obama for his extraordinary efforts to strengthen international diplomacy and cooperation between peoples.”

Because the reference is not referring to the text the user had initially intended, the transcluded document becomes less useful.
Some of the examples of implementation of transclusion using static referencing are as follows:

- Gupta, V. (August 2008). "Fine-Grained Addressability to Support Large-Scale Collaborative Document Development"
- Di Iorio, A., & Lumley, J. (July 2009). "From XML Inclusions to XML Transclusions"

Such implementations can only be employed when the user is sure that the source page is not going to change, but this takes away one of the major advantages of transclusion.

The researcher’s objective is to suggest a solution for this problem and come up with a prototype that would work even for dynamic pages.

**Background and Significance**

The fundamental difference between Nelson's vision and the Web today is that hyperlinks refer to ‘resources’ while his transclusions reference ‘representations.’

A representation may change through the course of time, but the resource could still be the same. A ‘transcluded’ document contains reference to the original text, not to the text itself. So if the source changes, the change is automatically reflected in the final output. This ensures the reader that this new ‘transcluded’ document always has access to the newest information. Moreover, transclusion would allow readers to access the original source very conveniently, and also to find out the context in which it was originally used.
Although this was the original vision for the Web, for simplicity the entire concept of transclusion was replaced by hyperlinks.

“HTML is precisely what we were trying to PREVENT—ever-breaking links, links going outward only, quotes you can't follow to their origins, no version management, no rights management.” – Ted Nelson

One of the primary reasons transclusion was replaced by something that was much simpler was the limited network bandwidth and processing power of computers in those days. Now, with the ever-increasing processing powers, this concept has re-emerged, and can benefit the publishing industry immensely by presenting information to users much more comprehensively and effortlessly.

**Interest of Study**

The researcher was working with Open Publishing Lab, RIT, as a graduate assistant. During his work, the researcher worked on a publishing platform that would enable users to gather content from the Web, and to unify and publish it to multiple formats. It was during this work that the researcher came across this concept and realized how easy it would be for users to gather content from the Web and publish their work. This was the primary interest for the researcher on this topic. After detailed study on transclusion, and how other researchers have implemented it, the researcher believes that he has a solution to eliminate the issues involved in transcluding text from dynamic source pages. Thus, the researcher wants to test his ideas and design, and to test a prototype as a proof of
concept, hoping that this will help the publishing industry, and more importantly, young, fresh authors to express themselves.
Chapter Two

Review of the Literature

Introduction

Any document is a collection of ideas created by human minds. Hence, the purpose of a document is to enable its authors to present their ideas effectively, such that readers can comprehend the ideas presented in it, and re-use them as a framework for establishing their own ideas. Thus one of the primary functionality that a document should provide is easy re-use of data present in it (Nelson, 2007).

In order to quote part of a document, people usually copy and paste the existing data into their new document (Krottmaier, & Maurer, July 2001). This results in a loss of connection between the source document and the destination document, and thus propagation of ideas from one document to the other is not effective (Nelson, 2007).

The reader of the new document might not be able to comprehend the quoted text and its importance in the new document unless the reader has access to the original document to understand the context in which the quoted text was originally used.

Many researchers have proposed and implemented solutions to handle the issue of lost context. The following sections will talk about some of the suggested solutions for
digital/Web content, the technologies that were used by them, and some of the issues that
still need to be handled to achieve re-use of content/ideas from dynamic documents.

**What is Transclusion?**

Transclusion is a form of smart inclusion that allows an author to include other authors’
work, not by duplicating the original work, but by including a reference to it
(Nelson, 1995).

For example:

Say there is a document with content that reads, “The quick brown fox jumps over the
lazy dog is a sentence that uses every character of the English alphabet.” Call it
Document A.

Now, an author wishes to create another document in which the author wants to quote a
portion of Document A. The author wants the new document to read, “The quick brown
fox jumps over the lazy dog is a very interesting sentence.”

In order to do so, the author can copy the required portion from Document A and then
paste it in the new document and append the rest of the content to it. In this case the
connection between this new document and Document A is lost, and there is a
duplication of bytes.
The author can do the same thing using the concept of transclusion. In the new document the author can refer to the quoted text by writing:

<transclude text from source=DocumentA, startindex=1, number_of_characters=43> is a very interesting sentence.

This results in a new document with content that reads, “The quick brown fox jumps over the lazy dog is a very interesting sentence.” This also ensures that the connection between the two documents is maintained and the context is not lost (Nelson, 2007).

For this to work, an editor or viewer must identify that the document contains some transcluding text and can retrieve the required text from the source document.

Features, such as highlighting of the transcluding text and side by side comparison of the original document and the transcluded document, can also be added to enable authors to present the information comprehensively (Kolbitsch, & Maurer, 2006).

It can be observed that the new document does not contain the actual text, but rather a reference to the quoted text.

Any changes in the source document are automatically reflected in the destination document, thereby ensuring that the new document always contains the latest information (Nelson, 2007).

The example presented above is the simplest form of transclusion; and many existing technologies like URLs, DOM structure, XML, AJAX, basic HTML tags, etc., can be used to develop an environment that allows authors to quote text from remote documents.
Several such environments, and the technologies used to develop them, are discussed below.

**Existing implementations of Transclusion**

Using basic HTML Tags

The most crucial part for implementation of transclusion is a way to refer to a resource.

Any page/object has a unique address associated with it. These addresses are called URLs or Universal Resource Locators. Thus, any page/object can be uniquely referred to using these URLs (Lee, 1994).

A URL looks like: ‘http://www.w3.org/Addressing/rfc1630.txt’

Using the HTML specifications, there are several tags that are used for implementation of transclusion.

<iframe>: Using this tag an inline frame containing a remote resource can be included in the existing page.

    <iframe src="source_document.html">
    </iframe>

Using this tag in a HTML page, the author can make reference to remote documents.

When the HTML is viewed in an HTML browser, the browser fetches the source document from the URL specified in the ‘src’ attribute and displays the entire content of the source page (Raggett, 1999).
This ensures that there is a connection between this new HTML page, and the source page. Moreover, the source content is not copied into this new HTML document, but only makes reference to it.

The problem with this is that a reference to a document fragment cannot be made, as the entire source page would be displayed, and not just a part of it (Kolbitsch, & Maurer, 2006).

<object>: Unlike the iframe tag, where only HTML pages could have been included, the object tag can be used to include objects such as images, audio, videos, Java applets, ActiveX, PDF, and Flash animations (Raggett, 1999).

Again, it is not possible to include a portion of any object, only the whole object.

These tags do reflect an idea of transclusion, though very limited in nature. Moreover, these tags do not support transclusion of text.

An amendment to the HTML specification was suggested, which proposed that a new tag <text> be added. The main attributes of this tag would be the source URL, the starting index, and the length of the text to be transcluded. This would also enable transclusion of document fragments (Pam, 1997).

Although the proposal seems quite promising for the implementation of transclusion on the Web, it has not been accepted, and none of the browsers have implemented this feature (Kolbitsch, & Maurer, 2006).
Thus, it can be observed that HTML does provide ways for transcluding information from remote documents, but in a very limited way. Document fragments are not handled and it is not possible to carry out text-based transclusion.

Using Document Object Model (DOM) Nodes

Josef Kolbitsch, & Hermann Maurer designed a prototype to implement transclusion in a HTML-based environment. In their design transclusion of document fragments was handled very well, and fragments could be of any size from a single character to the entire content of a page (Kolbitsch, & Maurer, 2006).

The other interesting feature of their implementation was that technologies they used included HTML, DOM, Java Script, and HTTP. These are supported by every browser on the Web; thus their implementation requires no special software or plug-ins.

DOM is a language-neutral interface that defines a standard way of accessing, and updating the content, structure, and style of a document. DOM presents a document in a tree structure. (www.w3.org)
The following set of examples will be used throughout this thesis to explain the various technologies that have been used by other researchers to implement transclusion.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>This is the source document.</td>
<td>&lt;div&gt;</td>
</tr>
<tr>
<td>Paragraph 1 of source document.</td>
<td>&lt;h&gt;This is the source document&lt;/h&gt;</td>
</tr>
<tr>
<td>Paragraph 2 of source document.</td>
<td>&lt;div&gt;</td>
</tr>
<tr>
<td>End of source document.</td>
<td>&lt;p&gt;Paragraph 1 of source document&lt;/p&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;p&gt;Paragraph 2 of source document&lt;/p&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/div&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;h&gt;End of source document&lt;/h&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/div&gt;</td>
</tr>
</tbody>
</table>

The DOM representation of the above source HTML document would be a tree structure and would look like the following:

![DOM structure of the above HTML document](image)

Figure 1: DOM structure of the above HTML document
All of these boxes are termed *nodes*. The one at the top is the root node of this DOM tree. The data present inside each of the nodes are either normal text that appears in the text document, or are the tag names of the node (<h>, <div>, <p>). The nodes containing the text are termed *text nodes*.

Any implementation of transclusion involves two operations:

1) Creation of transclusion (when some text from a source page is quoted)
2) Retrieval of transcluded text (when the page containing transclusion is viewed)

Whenever a user is quoting text, a JavaScript is called, and based on the selection the user has made, the script evaluates the start point and the end point of the selected text (Kolbitsch, & Maurer, 2006).

The parameters that Kolbitsch, & Maurer implementation uses are as follows:

<src>: URL of the source page
<atag>: denotes the name of the tag in which transclusion starts
<ftag>: denotes the name of the tag in which transclusion ends
<aindex>: denotes the index of the tag in which transclusion starts
<findex>: denotes the index of the tag in which transclusion ends
<aoffset>: denotes the offset within the start tag
<foffset>: denotes the offset within the end tag
For the purpose of illustration, a standard explanation would include a set of examples and their respective diagram using DOM structure.

Example:

The DOM of the HTML source is:

![Figure 2: DOM structure of the source HTML document](image)

```
<transclusion src=URLofSourceDocument
  atag="h" aindex="0" aoffset="0"
  ftag="p" aindex="1" aoffset="11" />
```

This means that the user has selected text that has a starting point inside the first `<h>` tag beginning with the first character, continuing to the second `<p>` tag, and ending with the 12th character.
This type of reference ensures that document fragments of any size can be quoted.

When retrieving the transclusion, another JavaScript is called that fetches the source page and evaluates the selection based on the transclusion parameters.

Since the reference is static, if the contents of the source page changes, the transclusion fails. To avoid this, this implementation also stores a fingerprint/hash/encryption of the content of that page. Thus, if the source page changed, the new fingerprint would not match the stored fingerprint, and the implementation would know that the page had changed and would flag a warning. (Kolbitsch, & Maurer, 2006)

This implementation takes care of document fragments very well, but when the source page has changed, it just flags a warning. Thus, this implementation does not really take care of transcluding text from dynamic pages.
**Using Asynchronous JavaScript and XML (AJAX)**

AJAX is a Web development technique that is used to create interactive applications on the Web. The technique allows for any page on the client side to be updated without the need to reload it manually. (Garrett, 2005).

An XMLHttpRequest object can be used to retrieve any source page, and can be displayed on the client page (www.w3.org, 2009).

As mentioned in the previous section, implementation of transclusion involves two major actions.

In AJAX, reference to a document fragment of a page is done using DOM. Thus, in the first action that involves creation of transclusion, DOM must be used to refer to the selection. The start point and the end point are located and represented in an identical way, as Kolbitsch, and Maurer suggested (using atag, ftag, aindex, findex, aoffset, foffset).

AJAX enables retrieval of transcluded text more interactively. The document containing the transcluded text updates itself as the source page changes without the user having to reload the document.

Although AJAX with DOM does not really help transclude text from a dynamic page, if referencing from a dynamic page is possible, such that the reference automatically updates when the source page changes, AJAX would make the transcluded document very interactive.
Using Extensible Markup Language (XML)

XML is a specification that defines a way to encode documents for the Web. XML provides a number of features that can be used to create documents composed of fragments from multiple sources. (http://www.w3.org/XML/)

W3C Standards like XInclude, Xlink can be used for writing inclusion directives to include remote documents in the user document (Derose, 2001).

Example:

The DOM representation of the HTML source file:

Figure 4: DOM structure of the source XML document
Let the user document be (HTML representation):

```
<div>
  <h>user Document</h>
  <p>Paragraph 1 of user document</p>
  <transclusion src="sourcedoc"/>
  <p>Paragraph 2 of user document</p>
</div>
```

DOM representation of the user document:

Figure 5: DOM structure of the user XML document with transclusion tag

In Figure 5 the user document contains quoted text from ‘sourcedoc’.

The transclusion tag would fetch the entire content of ‘sourcedoc’.
The DOM of final generated user document would be as follows:

![DOM structure diagram](image)

Figure 6: DOM structure of the generated user document (including transcluded fragments from the source document)

XPath can be used to select a specific tag from the source document; and if the path of that specific tag were provided, only the contents in that tag would be fetched (DuCharme, 2003).

Example of the user document where the path of a specific tag is provided:

Say the user wants to quote the heading and the second paragraph from the source document. The user document would look like this:

```xml
<div>
  <h>user Document</h>
  <p>Paragraph 1 of user document</p>
  <transclusion src="sourcedoc.xml" xpath="//div/h"/> (This would fetch just the first heading.)
  <transclusion src="sourcedoc.xml" xpath="//div/p[2]"/> (This would fetch just the second paragraph.)
</div>
```
<p>Paragraph 2 of user document</p>
</div>

DOM representation of the User Document:

Figure 7: DOM structure of the user XML document using the XPath attribute inside the transclusion tag

This would result in a DOM that would look like this:

Figure 8: DOM structure of the generated user document (corresponding to Figure 7)

It is important to note that the transcluded portions of the source document are not actually added to the user document, but the DOM of the user document is dynamically
generated every time it is viewed; and the user document still contains only the reference
to these portions from the source document.

The granularity, i.e., the minimum amount of text that can be transcluded using this
method, is content of any tag. It is not possible to address any fragment inside any tag
using this method.

The most interesting feature of this implementation is that XInclude and XPath can
handle recursive transclusion. That is to say, if the source document itself contains some
transcluded text, that transcluded text would also be fetched along with the desired
contents from the source page (Di Iorio, & Lumley, 2009).

Thus, XML provides very powerful tools for the implementation of transclusion.
Moreover, it does not require any special plug-ins to be installed, as XML is supported by
all major browsers. To view a transcluded document, all the user has to do is to open it in
the browser, and the browser takes care of fetching the desired document fragments from
the remote documents (Wilde, & Lowe, 2002).

Since XML uses XPath to refer to different nodes, even if some text is added or modified
in that node, the changes are reflected in the transcluded document. So it does offer
solutions to dynamic source pages. But the solution is very limited in nature, since it only
works if the text is modified just inside the node. If another node were added to the page,
then this implementation would fail to refer to the original text (Di Iorio, &
Lumley, 2009).
Example:

Let’s say the text inside the second paragraph of the source document was changed.

The DOM representation of the source document would be as follows:

![DOM structure of the source document](image)

Figure 9: DOM structure of the source document (2nd paragraph changed)

The DOM representation of the final user document would look like this:

![DOM structure of the generated user document](image)

Figure 10: DOM structure of the generated user document reflecting the change in Figure 9
The change in the source document has been reflected since:

Figure 11: Transclusion nodes and the document fragment to which they refer

Thus, we see that the change in the source document is reflected in the user document.

But if the source document is modified as shown below:

Figure 12: DOM structure of the source document (change: paragraph added between 1 and 2)
The DOM of the user document would look like this:

![DOM structure of the generated user document reflecting change corresponding to Figure 12](image)

This is because:

![Transclusion nodes and the document fragment to which they refer (corresponding to Figure 12)](image)

Thus, the transcluded text is no longer the text that the user initially selected.

Even with so many advantages, XML itself does not provide for a solution to transclude text from dynamic pages.
Purple Number – Node Identifier

Purple number is a way to uniquely identify an element/tag/node in a HTML/XML document. The basic idea is that a unique number is assigned to any text node, and this number is not based on the content or its hierarchical location in the DOM tree (Kim, 2003).

In the previous example, using XML, a node was uniquely identified by its path, i.e., its hierarchical location in the DOM tree. Thus, if any node was added to the tree, the previous path might no longer refer to the node.

Purple numbers, also termed as Node Identifiers (NID), are unique to the HTML page. If the node’s contents were modified, or its hierarchical position changed, the number would still be the same, and thus could be used to refer to the node (Gupta, 2009).

Thus, Purple numbers can be used quite effectively to implement transclusion since, unlike other implementations, the number still refers to the same node the user initially selected, irrespective of changes made in that node, or in any other node. This would enable transclusion of document fragments from a dynamic source page.

When writing an HTML document, an author can also add an ‘id’ attribute to each of the nodes. With these ‘id’ attributes, the nodes can also be uniquely identified. NID numbers work in the exact same fashion, and instead of the author having to write id attributes for each of the tags, a small collection of tools named ‘Purple’ can be used to add these
purple numbers, which then serve as anchors attached to each of the nodes (Blue Oxen Associates, 2007).

Adding these unique identifiers (either manually by using the ‘id’ attribute or by using ‘Purple’) must done by the author of the original page. So if the source page does not contain these unique identifiers, transcluding text from those documents is not possible.

Although the concept of Purple numbers is very promising, especially for implementation of transclusion, authors cannot be expected from to generate these unique identifiers, and there are very few HTML pages on the Web that actually use these Purple numbers.

**Ideal Implementation of Transclusion**

According to Ted Nelson, the major flaw in the current state of World Wide Web is that the links are unidirectional. This makes the references from the user page to the source page unidirectional as well. Thus, if the source page changes, the user document has no idea about those changes, and thus the references are not of much use. Consider a situation where all these references are bi-directional. If this were possible, the source page would have knowledge about all the other user pages that contain quoted text from the source page. It would also have knowledge of the start and the end of each selection. With this knowledge, if the source page changed, it would automatically notify all the user pages, and provide them with a new set of transclusion parameters that would identify the selection users had initially made (Nelson, 2007).
In Project Xanadu, Ted Nelson’s original hypertext system, transclusion was implemented in a very similar way. In that implementation both the document model, and the way a document is referred to, are very different from what is available today (Nelson, 1987).

The project demonstrates the effectiveness of transclusion and shows what a document has the potential to be compared to what it is now.

This ideal state of Web would make it very easy for people to exchange and re-use ideas, which, in turn, would enable them to establish an effective, collaborative content-generation platform.

Summary

Transclusion has been implemented by many researchers from around the globe using various technologies and tools. All these implementations vary in their simplicity of design, the prerequisites required for their implementation, and the additional features they provide.

The implementation by Josef Kolbitsch and Hermann Maurer, where only DOM tree was used, was effective, simple in design, and displayed what could be achieved using transclusion. The implementation using AJAX and DOM tree made the transcluded document interactive, and thus very useful when transcluding text from dynamic websites where content changes are very frequent. The implementation using XML and DOM was very powerful, as it could handle recursive transclusion. Usage of Purple Numbers for
implementation of transclusion was really promising, as it could handle transclusion from dynamic pages conveniently and yet very effectively. But this implementation required the original authors to associate unique numbers to each of their HTML nodes.

All of the above implementations could be used for different applications based on their requirements. In most of the implementations (except when using Purple Numbers) transcluding text from dynamic source pages was not supported. Based on the existing state of the Web, if this issue could be handled, transclusion would be much more effective, and the Web could reach a state closer to what Ted Nelson originally imagined.
Chapter Three

Research Question

Develop an algorithm based on encryption of text that can be used to transclude text from
dynamically changing source pages, such that the transcluded document would accurately
reflect the changes made in the source document.
Chapter Four

Methodology

Design of Algorithm

An algorithm was required that would ensure that changes in the source document would automatically and accurately reflect in the transcluded user document.

The basic requirement for transclusion is that the user document should not include the transcluded text, but only contain some form of reference to the document fragments of the source page.

Overview of the Algorithm

The algorithm was based on encryption of the transcluded text and its positional relationship. When retrieving the transcluded text from the source document the source document was encrypted using the exact same encryption technique, and then the encrypted transcluded text was compared with the encryption of the source document. The fragment of the source document whose encryption matches closest to the encrypted transcluded text was returned as the transcluded text.
Step-by-step Description of Algorithm:

The United States Declaration of Independence is a statement adopted by the Continental Congress on July 4, 1776, which announced that the thirteen American colonies then at war with Great Britain were now independent states, and thus no longer a part of the British Empire. Written primarily by Thomas Jefferson, the Declaration is a formal explanation of why Congress had voted on July 2 to declare independence from Great Britain, more than a year after the outbreak of the American Revolutionary War. The birthday of the United States of America is celebrated on July 4, the day the wording of the Declaration was approved by Congress.

After finalizing the text on July 4, Congress issued the Declaration of Independence in several forms. It was initially published as a printed broadside that was widely distributed and read to the public. The most famous version of the Declaration, a signed copy that is usually regarded as the Declaration of Independence on display at the National Archives in Washington, D.C. Although the wording of the Declaration was approved on July 4, the date of its signing has been disputed. Most historians have concluded that it was signed nearly a month after its adoption, on August 2, 1776, and not on July 4 as is commonly believed.

Figure 15: Original source document

Figure 16: Initial selection in the source document

Figure 17: Changed source document (highlighted portion indicates the new addition)
Step 1: Encryption of Transcluded Text: The string was first split into an array of words. Each of these words was then encrypted in the following manner:

- Each of the letters in a word was converted into its integer equivalent (a->1, b->2, …, z->26).
- Each of these integers was then added to its positional index in the word, and the sum was then multiplied with its positional index. (In the word “hello” the letter ‘h’ is converted to \[ (8 + 1) \times 1 \] = 9; the letter ‘o’ is converted to \[ (15 + 5) \times 5 \] = 100).
- Thus, each alphabet in the word was represented by an integer.
- The sum of all these integers was taken, and this sum was used as the encryption of the word.

Example:

Encryption of the word “hello” would result in \[ (8+1)*1 + (5+2)*2 + (12+3)*3 + (12+4)*4 + (15+5)*5 \] = \[ 9 + 14 + 45 + 64 + 100 \] = 232.

- Thus, a number represented a word. Each of the words in the string went through the same encryption process, and finally the string was represented by an array of numbers.
- A final number was added to this array of numbers, which was the total length of the transcluded text.
- This array of numbers, along with the location of the source file, was used as the reference to the transcluded text in the user document.
Step 2: Encryption of Source Text: Using the link to the source file, the source file was read and stored in a string. This string was encrypted in an exact fashion as described in Step 1. The position of each word was also calculated and stored along with the encryption.
Step 3: Matching the Encryption of Transcluded Text with the Encryption of the Source Text:

a) The transcluded text and the source text were both represented by an array of integers after Steps 1 and 2.

Figure 20: Encrypted transcluded text corresponding to Figure 16

```
<table>
<thead>
<tr>
<th>Step 3: Encrypted Transcluded Text</th>
<th>Source Text</th>
</tr>
</thead>
</table>
```

Figure 21: Encryption of the new source document (the numbers in the brackets indicate the position of the corresponding word)

b) The Longest Consecutive Substring (LCS) in the two arrays was found using the standard algorithm to find LCS. (http://www.itl.nist.gov/div897/sqg/dads/HTML/longestCommonSubstring.html)
c) Using the positional information of words present in the source file, a start point was determined. In the above example, the first word that matches starts from position [18], and the last word that matches starts from position [267]. All the contents between these positions were selected as the initial message. The initial score was calculated by dividing the number of words that were found by the total number of words present in the selection.

\[
\text{Initial Score} = \frac{\text{NumberFound}}{\text{TotalNumberInSelection}}
\]

Figure 23: Starting point found along with its score

d) All the words (integers in the array) that were found in the previous steps were replaced by very high random numbers. This was done so that upon further execution of the LCS algorithm these words would not be picked up.
e) Once these steps were completed, the LCS algorithm was executed again.

Basically, this returned the $i^{th}$ largest substring, where $i$ is the number of times the LCS algorithm had been executed.

```
Encrypted New Source Text: 33[0], 254[1], 336[11], 1119[16], 26[30], 1004[33], 146[41], 767[54], 306[61], 63[69], 53[72], 112[75], 648[86], 49[97], 190[100], 163[114], 512[129], 229[130], 53[135], 544[139], 428[145], 572[157], 117[166], 48[171], 81[174], 143[178], 187[183], 336[189], 117[191], 119[192].
```

f) Again, using the positional information, part of the text located in this iteration was extracted from the source file. In Figure 25, all of the words between position 688 and 905 are extracted.

```
```

Figure 24: After replacing the words that were found in the previous step
Figure 26: Document fragment in the source corresponding to the match in Figure 25

g) The portion of the text that was found in the previous step is combined with the text that was previously found, along with the text in the source file that existed between the two.

Total text found in this iteration: Declaration of Independence is a statement adopted by the Continental Congress on July 4, 1776, which announced that the thirteen American colonies then at war with Great Britain were now independent states, and thus no longer a part of the British Empire. It's a declaration of the rights of man, particularly the second sentence, a sweeping statement of human rights: We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable Rights, that among these are Life, Liberty and the pursuit of Happiness.

This sentence has been called "one of the best-known sentences in the English language". Written primarily by Thomas Jefferson, the Declaration of Independence was adopted on July 4 to declare independence from Great Britain, more than a year after the outbreak of the American Revolutionary War.

Figure 27: Combined document fragment

h) The score was calculated as follows:

\[
\text{score} = wt \times \left( \frac{nwft}{tnwt} \right) + (1-wt) \times \left( \frac{tlt}{ltf} \right) \quad \text{[for } tlt \leq ltf\text{]}
\]

\[
wt \times \left( \frac{nwft}{tnwt} \right) + (1-wt) \quad \text{[for } tlt > ltf\text{]}
\]

Where,

wt = Constant weight term

nwft: Number of words found in transcluded text

tnwt: Total number of words in transcluded text

tlt: Total length of transcluded text

ltf: Length of text found
The score increased if the words in the transcluded text were found in the source document, but was reduced if, in order to find those words, the gathered text became very large. This ensured that the words found in an iteration of LCS implementation were closer to the portion of the text that was previously found, and were thus more likely to be part of the transcluded text that the user had initially selected. The value of constant weight (wt) determined the amount of penalty (reduction in score) given when the gathered text became large compared to length of the original transcluded text.

In the example above ‘wt’ was selected as 0.7, and the score was found to be:

<table>
<thead>
<tr>
<th>Total number of words found</th>
<th>Total number of words in original selection</th>
<th>Total length of text found</th>
</tr>
</thead>
<tbody>
<tr>
<td>73</td>
<td>73</td>
<td>888</td>
</tr>
</tbody>
</table>

Score : 0.87256

Figure 28: Score computed based on the document fragment corresponding to Figure 27

Basically, this means that all the words that were present in the original transcluded text (73) were located. But since some text was added in the source page, the total length of the new transcluded text became 888 instead of 482 (that was originally present). Using the formula as described above, the score turned out to be 0.87256.

i) Once the score was calculated, it was compared to the previous score, and if the new score was found to be greater than the previous score, and if some words from the original transcluded text were still missing, the algorithm went back to step ‘e’, and performed the same steps again.
j) If all of the words were found, and the score was greater than the previously-found score (as in the above example), the text found was returned as the new transcluded text. If the score found turned out to be less than the score that was found in the previous iteration, the text that was gathered in the previous iteration was returned.

Figure 29: Final document fragment returned

Finding the optimum value of ‘wt’:

The value of ‘wt’ selected in order to find the score was essential to ensure that the predicted final transcluded text accurately incorporated the changes made to the source. In order to find an optimum value, the algorithm was tested with different values of ‘wt’ and the text returned was analyzed for its accuracy. The value of ‘wt’ was selected such that the average accuracy would be highest for different scenarios (depending on the length of the source document, length of the transcluded document fragment, percentage change, number of changes, and the nature of changes performed on the source document).
Pseudo Code of the Main Algorithm:

1) Encrypt the transcluded text.

2) Encrypt the source text.

3) Find the longest substring between the encrypted transcluded text, and the encrypted source text.

4) Based on the longest substring found, assign a score.

5) Again, find the $i^{th}$ longest substring (where ‘$i$’ is the number of iterations).

6) Combine the substring found in this iteration to the portion of the text already found.

7) Assign a score to this new portion of text.

8) If the score is greater than the previously-found score, and there are still words to be found, go back to step 5.

9) When the maximum score is found, stop the loop and return the portion of text corresponding to that maximum score.

Analysis of the Algorithm

Since the accuracy of the returned transcluded text depended on the original source file, the portion of the text transcluded from the source file, the amount, number, and nature of changes in the source file, an automated script was written to test the accuracy of the algorithm. The steps that were taken are as follows:
1) A collection of 839 documents was downloaded from the Web. 
(http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes/20_newsgroups.tar.gz)

2) The script selected one file at a time, and treated it as a source file.

3) It then randomly selected a portion of that file as the transcluded text.

4) A random number between zero and two was chosen. This number denoted the number of additions to the original source file to be performed.

5) For each addition a random file was selected, and a random portion of this new file was picked and inserted at a random point in the original source file. The source file, with this addition, was treated as a new source file. Depending on the number of additions to be performed, new text was added onto this file.

6) A random number between zero and two was chosen again. This number denoted the number of deletions to the source file.

7) For each deletion a random point in the source file was selected, and a portion of the text was deleted. Portions of text were further deleted from it based on the number of deletions.

8) Since the script knew the changes that went through the original source file and the original transcluded text, the script calculated the expected transcluded text.
9) The new source file and the previous transcluded text was provided to the algorithm; and the algorithm returned the most probable transcluded text that reflected the changes performed on the source file.

10) This predicted transcluded text was then compared to the expected transcluded text, and thus, the accuracy of the algorithm for that situation was recorded by dividing the length of Longest Common Substring between the expected transcluded text and the predicted transcluded text, by the length of expected transcluded text.

11) The script recorded the percentage of change to the source document, number of deletions, number of additions, length of the source document, length of the original transcluded text, and the accuracy of the algorithm for that situation.

12) The script did the same for all documents.

13) The script was executed several times for all of the documents, just to ensure that the accuracy of the algorithm has been recorded for most different sizes of the source document, different sizes on the portions transcluded from the source documents, and different amounts, numbers, and the nature of changes in the source document.
Chapter Five

Results

An algorithm was written that returned transcluded text from dynamic source documents which accurately reflected the change in the source. In order to test the algorithm, a script was written that recorded the size of the source document, percentage of transcluded text, number of changes, percentage change, and the nature of changes that were performed on the source document. Based on each such situation, the accuracy of the algorithm was recorded.

This section includes the presentation of the results obtained from the test, and the significance of each result.

Percentage Change

Before the accuracy of the algorithm is presented it is important to understand the percentage changes of the source documents. The researcher hypothesized that the greater the change, the less effective the algorithm would be. The following result shows the percent changes that the original set of source document went through.
Each of the source documents was changed randomly. The percentage change for each document was recorded. Figure 30 shows the number of documents vs. percentage change. For example, 238 documents went through a change of less than 10%, and 51 documents went through a change that was greater than 50%, but less than 60%.

The percentage change was calculated using the following formula:

\[
\%\text{Change} = \frac{\text{Number of characters added} + \text{Number of characters deleted}}{\text{Total number of characters in the original source document}}
\]
Average Accuracy vs. Weights Selected in the Formula to Calculate the Score

The score indicates the quality of the transcluded text that is returned after execution of the algorithm. The algorithm assigned a score to a document fragment based on its probability of being the transcluded text that reflected the change in source document. Thus, the most important task for the algorithm was to find a document fragment that had the highest score.

As discussed in the Step 2h, the score was determined using the following formula:

\[
\text{score} = wt \times \frac{nwft}{tnwt} + (1-wt) \times \frac{tlt}{ltf} \quad \text{[for tlt} \leq \text{ltf]}
\]

\[
= wt \times \frac{nwft}{tnwt} + (1-wt) \quad \text{[for tlt} > \text{ltf]}
\]

Where,

\( wt \) = Constant weight term

\( nwft \): Number of words found in transcluded text

\( tnwt \): Total number of words in transcluded text

\( tlt \): Total length of transcluded text

\( ltf \): Length of text found

The parameter ‘wt’ was used to determine the penalty that should be given to the score if the length of the text found was greater than the length of the original transcluded text.

The value of the parameter ‘wt’ became more important when the insertions and/or deletions were performed in the source document such that insertion/deletion points were closer to the edge of the document fragment that was originally selected. The edge of a
document fragment generally would be the first or last couple of sentences in the document fragment. So if the insertion or deletion point was located somewhere within the first or last couple of sentences, then the value of ‘wt’ would play an important role. The following paragraph explains how the value of ‘wt’ determines the accuracy of the algorithm when the deletion point lies closer to the edge of the document fragment.

In the case of deletions, the algorithm should not fetch text with a length longer than the original transcluded text when searching for terms that were originally present in the source document, but have since been removed. For such situations a higher penalty should be given if the length of returned transcluded text was found to be greater than the length of original transcluded text. Thus the value of ‘wt’ should be kept low for such scenarios.

Example:

Here, a portion of the text has been deleted near the beginning of the originally-selected document fragment.

---

New Source:

The Declaration adopted by the Continental Congress on July 4, 1776 announced the American colonies are no longer a part of the British Empire. Its stature grew over the years, particularly the second sentence, a sweeping statement of human rights:

We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable Rights, that among these are Life, Liberty and the pursuit of Happiness.

This sentence has been called "one of the best-known sentences in the English language".

Written primarily by Thomas Jefferson, the Declaration is a formal explanation of why Congress had voted on July 2 to declare independence from Great Britain, more than a year after the outbreak of the American Revolutionary War. The birthday of the United States of America—Independence Day—is celebrated on July 4, the day the wording of the Declaration was approved by Congress.

After finalizing the text on July 4, Congress issued the Declaration of Independence in several forms. It was initially published as a printed broadside that was widely distributed and read to the public. The most famous version of the Declaration, a signed copy that is usually regarded as the Declaration of Independence, is on display at the National Archives in Washington, D.C. Although the wording of the Declaration was approved on July 4, the date of its signing has been disputed. Most historians have concluded that it was signed nearly a month after its adoption, on August 2, 1776, and not on July 4 as is commonly believed.

---
When the value of ‘wt’ was kept higher than 0.75, the final transcluded text returned was found to be:

\[
\text{Weight : 0.8} \]

\[
\text{Predicted final transcluded text :} \\
\text{----------------------------------------} \\
\text{adopted by the Continental Congress on July 4, 1776 announced the American colonies are no longer a part of the British Empire. Its stature grew over the years, particularly the second sentence, a sweeping statement of human rights: We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable Rights, that among these are Life, Liberty and the pursuit of Happiness. This sentence has been called 'one of the best-known sentences in the English language', written primarily by Thomas Jefferson, the Declaration is a formal declaration of why Congress had voted on July 2 to declare independence from Great Britain, more than a year after the outbreak of the American Revolutionary War. The birthday of the United States of America?Independence Day?is celebrated on July 4, the day the wording of the Declaration was approved by Congress. After finalizing the text on July 4, Congress issued the Declaration of Independence in several forms. It was initially published as a printed broadside that was widely distributed and read to the public. The most famous version of the Declaration, a signed copy that is usually regarded as the Declaration of Independence, is on display at the National Archives in Washington, D.} \\
\text{----------------------------------------} \\
\]

Figure 32: Returned transcluded text (‘wt’ = 0.8) corresponding to Figure 31

This was because, the value of ‘wt’ being higher, a lower penalty was imposed due to increase in length. Thus, in order to find the phrase ‘Declaration of Independence’, which was present in the original selection, but not in the same location in the changed source file, the algorithm returned text of a longer length compared to what was intended. With this returned text, the algorithm did manage to find the phrase ‘Declaration of Independence’, but the returned text did not match the intended transcluded text, as it included a few sentences that were not originally selected. The returned text should have ended after the phrase ‘American Revolutionary War’.

By keeping the value of ‘wt’ lower, and thereby imposing a higher penalty, the result obtained was found to be:
adopted by the Continental Congress on July 4, 1776 announced the American colonies are no longer a part of the British Empire. Its stature grew over the years, particularly the second sentence, a sweeping statement of human rights.

We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable Rights, that among these are Life, Liberty and the pursuit of Happiness.

This sentence has been called "one of the best-known sentences in the English language".

Written primarily by Thomas Jefferson, the Declaration is a formal explanation of why Congress had voted on July 2 to declare independence from Great Britain, more than a year after the outbreak of the American Revolutionary War.

Figure 33: Returned transcluded text (‘wt’ = 0.65) corresponding to Figure 31

This was much closer to the intended transcluded text.

If some text was added closer to the edge of the document fragment that was initially selected, the algorithm should try to find portions of the document fragment that were present beyond the point of insertion, even though the resulting returned text would be longer in length, compared to the original selection.

Example:

The United States Declaration of Independence is a statement adopted by the Continental Congress on July 4, 1776. Its stature grew over the years, particularly the second sentence, a sweeping statement of human rights:

We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable Rights, that among these are Life, Liberty and the pursuit of Happiness.

This sentence has been called "one of the best-known sentences in the English language". The Declaration announced that the thirteen American colonies were no longer a part of the British Empire, and thus no longer a part of the British Empire.

Written primarily by Thomas Jefferson, the Declaration is a formal explanation of why Congress had voted on July 2 to declare independence from Great Britain, more than a year after the outbreak of the American Revolutionary War.

Figure 34: Changed source document with a portion added near the beginning
Here, a few sentences were added closer to the beginning of the initially-selected document fragment.

When the value of ‘wt’ was kept low, i.e., increasing the penalty if the returned text length was higher than the original selection, the result obtained was found to be:

Weight : 0.6

Predicted final transcluded text :

The Declaration announced that the thirteen American colonies then at war with Great Britain were now independent states, and thus no longer a part of the British Empire. Written primarily by Thomas Jefferson, the Declaration is a formal explanation of why Congress had voted on July 2 to declare independence from Great Britain, more than a year after the outbreak of the American Revolutionary War.

Figure 35: Returned transcluded text (‘wt’ = 0.6) corresponding to Figure 34

Thus, we see that the portion of text added to the source, along with the portion of the document fragment that was present before the point of addition and present in the original selection, was not returned.

But when the penalty was reduced, or the value of ‘wt’ was increased, the result obtained was found to be:

Weight : 0.8

Predicted final transcluded text :

Declaration of Independence is a statement adopted by the Continental Congress on July 4, 1776. Its structure grew over the years, particularly the second sentence, a sweeping statement of human rights: We hold these truths to be self-evident, that all men are created equal, that they are endowed by their Creator with certain unalienable Rights, that among these are Life, Liberty and the pursuit of Happiness. Written primarily by Thomas Jefferson, the Declaration is a formal explanation of why Congress had voted on July 2 to declare independence from Great Britain, more than a year after the outbreak of the American Revolutionary War.

Figure 36: Returned transcluded text (‘wt’ = 0.8) corresponding to Figure 34
In this case, the final transcluded text accurately reflected the changes made to the original source document.

But since it is not possible to guess where the insertion or deletion would take place, nor the nature of change that the source would undergo, the optimum value of ‘wt’ should be used that would give higher accuracy of transcluded text for most situations.

In order to find an optimum value of ‘wt’, the test script was executed several times, each time changing the value of ‘wt’. For each execution of the test script, a set of almost thousand data points were recorded. So the average accuracy obtained for each value of ‘wt’ includes scenarios with different source document sizes, different sizes of the portions transcluded from the source documents, and different amounts, numbers, and nature of changes in the source document.

The results are presented below:

![Average Accuracy vs 'wt'](image)

Figure 37: Average Accuracy vs. ‘wt’
In Figure 37, the lowest accuracy obtained was when ‘wt’ = 0.9. This was because when ‘wt’ = 0.9, there was hardly any penalty given because of the length of the text returned. Similarly, the average accuracy was low for ‘wt’ = 0.6, as it imposed a lot of penalty.

The rest of the values are closer to each other.

But since the result showed that for ‘wt’ = 0.75, higher accuracy was obtained, ‘wt’ = 0.75 has been used for further analysis. The researcher would further discuss the potential of using various values of ‘wt’ depending on other factors.

**Accuracy of the Final Transcluded Text Compared to the Expected Outcome**

The original source document, the original selection, and the changes performed on the source document were all recorded for each of the documents in the test set. Using all this information, the expected transcluded text was calculated.

The predicted final transcluded text was obtained by the researcher’s algorithm using the reference to the transclusion (this includes the location of source document and the encryption of the initial selected document fragment).

The predicted final transcluded text was compared with the expected transcluded text and the accuracy of the algorithm for that data set was computed.

The accuracy obtained for all of the documents in the test set was recorded.
The result obtained is as follows:

Figure 38: Histogram of average accuracy

Figure 38 describes the overall accuracy of the algorithm irrespective of the nature of changes and the percent of change that each of the documents went through.

The plot indicates that the predicted transcluded text returned from the algorithm matched exactly as the expected result for 492 of the 839 documents comprising the total test set. This indicates that for close to 60% of the test set, the algorithm returned the final transcluded text that matched the expected outcome exactly, i.e., the final transcluded text reflected all of the changes that were made to the original source document.

Figure 38 also shows that 173 times the algorithm returned transcluded text that matched the expected outcome at least 90%, but was not identical.
This shows that almost 80% of the results matched the expected outcome to at least 90%.

**Accuracy Percentage vs. Change Percentage to the Source Document**

The accuracy of the algorithm depended on the percentage of change the original source document has gone through. Figure 38 showed the overall accuracy of the algorithm for all the documents irrespective of the percentage of change to the source document.

Figure 39 shows how the algorithm performed with the varying change percentage that the source document had undergone.

Figure 39: Average accuracy vs. percentage of change

In Figure 39, we can see that when the change percentage was less than 10%, the average accuracy of the algorithm was almost 1, whereas when it was about 50%, the average accuracy of the algorithm was close to 0.9. Even when the source document was changed more than 100%, the average accuracy was still greater than 0.75.
It is important to note that the change could be anywhere in the source document, and not necessarily between the document fragment that was originally selected.

**Accuracy vs. Nature of Change that the Initially-Selected Document Fragment has Undergone**

In Figure 39, the accuracy of the algorithm was shown depending on the percentage of change to the original source document. The change percentage did not consider the nature and the number of changes that the document fragment in the original source went through.

Figure 40 displays the average accuracy of the algorithm depending on the nature and the number of changes that the document fragment went through. Even though the document fragment that was originally selected had not changed, it was possible that other portions of the original document could have changed.
The results are as follows:

Figure 40 indicates that the average accuracy of the transcluded text returned from the algorithm was 0.99 when number of additions and number of deletions to the initially-selected document fragment was equal to zero.

When the document fragment underwent only one change (either deletion or addition) the average accuracy was between 0.95 and 0.96. When the document fragment underwent more than one change the average accuracy gradually decreased, and finally came down to 0.72 when the document fragment underwent addition and deletions twice each.

Figure 40 also indicates that the nature of change (addition or deletion) had little effect on the accuracy.
A maximum of four changes (two additions and two deletions) were carried on the original source document for the analysis of the algorithm.
Transclusion is a concept that allows users to reuse document fragments from different source pages, not by duplicating it, but by including a reference to the original work. A ‘transcluded’ document contains reference to the original text, not to the text itself. So if the source changes, the change is automatically reflected in the final output. This ensures the reader that this new ‘transcluded’ document always has access to the newest information. Transclusion has been implemented by many researchers from around the globe using various technologies and tools. But most of their implementations made use of static reference to the document fragment being transcluded. Thus, if the source page changed, their implementations might not be able to retrieve the document fragment that was originally selected.

The objective of this research was to find an algorithm that could be used to transclude text from dynamically changing source pages, such that the transcluded user document would accurately reflect the changes made in the source document.

The algorithm explained in this thesis fulfills the required objective. In summary, from the results obtained, it was found that the accuracy of the algorithm depended on a
number of factors such as: percentage change of the source document, and number and nature of changes that the original source document undergoes.

A test set of 839 documents was created. A document fragment was selected randomly from each of the documents, and the encryption of the document fragment formed the basis of the transclusion reference parameters. Each of the documents was then changed randomly and the algorithm was run to test whether the accurate document fragment was returned based on the changes made in the source document.

It was found that overall the accuracy of the algorithm was 0.92. For about 60% of the test set, the returned document fragment matched exactly to the expected outcome reflecting all the changes made to the original source document. Moreover, for more than 80% of the test set, the predicted result matched more than 90% to the expected outcome.

It was observed that the accuracy of the algorithm decreased with an increase in the percentage of change to the source document. This result is intuitive as well, since the more the source document changes, the harder it is to predict the final transcluded text, as some changes might be left out, or some extra portions of the document might be returned. The algorithm turned out to be quite forgiving, as even when the source document had changed 100%, the average accuracy still turned out to be more than 75%.

The nature of change (addition or deletion) had little effect on the accuracy. On increase in the number of changes, there was a decrement in the average accuracy. Based on the results obtained, the algorithm performed well in most cases, and was ever forgiving as
the average accuracy was greater than 0.75 even when 100% of the original document was changed.

As a result this research can be used as a framework to transclude document fragments from dynamic source pages, while still ensuring that the changes made to the source page are reflected in the user document.

In the researcher’s algorithm the connection between the user document and the source document is retained, and the user document always has access to the newest information, thereby allowing users to present their ideas effectively, and re-use them as a framework for establishing their own ideas.

According to this researcher, transclusion is a very promising concept and should replace all tools for copy pasting, and cross-referencing. Unfortunately, transclusion has not been implemented by document creation applications like Acrobat, Word, etc. Simulating paper in document creation software may not harness the power of digital documents.

Moreover, transclusion can be used for collaborative document creation, where every author writes his own part, and transcludes fragments that other authors have contributed. This would make the collaborative document creation so much more simple and efficient.

There are several limitations to the concept of transclusion. If an author created a document and had quoted several other documents using transclusion, if any of the source documents changed, then the user document would change as well. And, if any of the source documents were removed or made inaccessible, the document fragment that was
transcluded from that source would get eliminated from the user document. Thus, without the choice of the user, his document would change, and even make it incomprehensible.

The electronic document using transclusions would not have a fixed state unlike its printed version. This can be avoided when all the changes made to the source documents are saved separately with a different version number by the authors of source documents, and all the previous versions remain unchanged and accessible. This would allow users to see the changes that have been made in the various revisions, and use the change that he or she believes is appropriate for his or her document. Obviously, the large number of versions may also be problematic when identifying a specific version.

Further, a document that is never really ‘fixed’ will create a number of archiving issues.

The publishing industry could benefit from this researcher’s work as dynamically transcluded documents would finally be able to fulfill their real objective, that is, ‘expression, reception, and re-use’ of ideas; and thus present information to users much more comprehensively and effortlessly.

**Limitations**

The algorithm discussed in this research would fail if there were insertions following the selected document fragment, representing a gradual accumulation of content, since the researcher’s algorithm would not be able to include the newly-added text.

If the source document contains repetitive text fragments with lots of common substrings, the researcher’s algorithm might fail to return the transcluded text that accurately reflects
the changes in the source document because the algorithm is based on finding common substrings between the transcluded text and the source.

The research involved developing only a prototype that can be used as a proof-of-concept that demonstrates the transclusion of text from dynamic pages. Thus, it was not a full-fledged software application with a graphical user interface. Additionally, the algorithm works only for text documents, and not for document formats such as HTML, DOC, PDF, etc.

**Recommendations for further research**

The researcher found that the accuracy of the algorithm depends on the value of the parameter ‘wt’, and the value of ‘wt’ should be adjusted based on the nature of the change performed on the source document. In the future, using the existing framework, if based on the phrases found in each iteration of LCS (longest common substring) and their respective positions in the changed source page, a method may be found to predict the nature of change that the original source document might undergo, and thus change the value of ‘wt’ dynamically, and not keep it as 0.75 for all cases (as the researcher did in this thesis). In that case, the accuracy of the algorithm should be increased.

Only text documents were used for this research. But the same principles can be used for other document formats. In the future, if the framework of the algorithm presented in this research is used for other document formats, then the use of transclusion from dynamic pages would be greatly increased, as most of the document formats that users work with are either HTML, PDF, and DOC.
Chapter Seven

Bibliography


Gupta, V. Fine-grained addressability to support large-scale collaborative document development. M.S. dissertation, Northeastern University.


http://www.w3.org/TR/XMLHttpRequest/

W3Schools. AJAX Tutorial. Retrieved October, 2009 from

http://www.w3schools.com/Ajax/Default.Asp