A Characterization of Compound Documents on the Web

Eyal de Lara[†], Dan S. Wallach[‡], and Willy Zwaenepoel[‡]

[†] Department of Electrical and Computer Engineering
[‡] Department of Computer Science
Rice University

Abstract

Recent developments in office productivity suites make it easier for users to publish rich *compound documents* on the Web. Compound documents appear as a single unit of information but may contain data generated by different applications, such as text, images, and spreadsheets. Given the popularity enjoyed by these office suites and the pervasiveness of the Web as a publication medium, we expect that in the near future these compound documents will become an increasing proportion of the Web's content. As a result, the content handled by servers, proxies, and browsers may change considerably from what is currently observed. Furthermore, these compound documents are currently treated as opaque byte streams, but future Web infrastructure may wish to understand their internal structure to provide higher-quality service.

In order to guide the design of this future Web infrastructure, we characterize compound documents currently found on the Web. Previous studies of Web content either ignored these document types altogether or did not consider their internal structure. We study compound documents originated by the three most popular applications from the Microsoft Office suite: Word, Excel, and PowerPoint. Our study encompasses over 12,500 documents retrieved from 935 different Web sites. Our main conclusions are:

1. Compound documents are in general much larger than current HTML documents.

2. For large documents, embedded objects and images make up a large part of the documents' size.

3. For small documents, XML format produces much larger documents than OLE. For large documents, there is little difference.

4. Compression considerably reduces the size of documents in both formats.

1 Introduction

Productivity tools, often part of "office suites," are the most popular applications for creating documents. Their popularity derives, to some extent, from their capability to create compound documents that include data from more than one application.

The documents produced by office suites have been, until recently, very unwieldy for putting on the Web. These documents can be quite large, sometimes hundreds of kilobytes, and Web browsers do not know how to display them without first completely downloading the document.

Recent improvements in office suites have made it easier to publish compound documents on the Web by exporting browser-compatible data types. These improvements, coupled with the popularity of productivity tools, will likely have a strong impact on the content of the Web and the infrastructure that supports this content. Previous studies have either overlooked compound document altogether or have treated them as black boxes [2, 18, 16, 1, 17]. Therefore, it becomes important to study and characterize compound documents on the Web, as they are now, to predict where the Web might be going, and how Web infrastructure should support compound documents in the future.

For this paper, we studied compound documents generated by the three most popular application of the Microsoft Office suite: Word, Excel, and PowerPoint. We chose to focus on Microsoft Office applications based on two factors. First, Microsoft Office is the most widely-used productivity suite. Moreover, a significant number of Microsoft Office documents are available on the Web, enabling us to gather the data for our experiments. Second, Microsoft Office 2000 supports two native file formats: the proprietary OLE-based binary format and a new XML format. By using Office 2000 to convert old files to the new XML format, we can compare the tradeoffs of using a proprietary binary-based file format against a modern standards-based text format, both as intermediate formats suitable for document editing, and as publishing formats, suitable only for reading.

We downloaded over 12500 documents, comprising over 4 GB of data, from 935 different sites. Our main results are:

1. Compound documents are in general much larger than current HTML documents. The tail of the size distribution follows the same power-log distribution observed with HTML documents.

2. For large documents, embedded objects and images comprise a large part of the document.

3. For small documents, XML format produces much larger documents than OLE. For large documents, there is little difference.

4. Compression considerably reduces the size of documents in both formats.

The rest of this document is organized as follows. Section 2 provides some background on compound documents and their enabling technology. We also discus relevant characteristics of the three Microsoft Office applications that we use in this study. Section 3 describes the documents we used in our experiments. Section 4 presents our experimental results. Finally, section 5 discusses our conclusions.

2 Background

2.1 Compound documents

To its user, a compound document appears to be a single unit of information, but in fact it can contain elements created by different applications. A compound document could, for instance, consist of a spreadsheet and several images embedded into a text document.

In the general case, every data type in a compound document (spreadsheet, text, images, sound, etc) is created and managed by a different application. The different applications used to create the document can be thought of as *software components* that provide services that are invoked to create, edit, and display the compound document.

2.2 Enabling technologies

2.2.1 Component standards

Compound documents result from combining the data created by two or more disjoint software components. As a result, there is a need for a standard to govern the interactions between components. Some of the most visible standards are COM/OLE [3, 4], SOM/OpenDoc [14], and JavaBeans [7]. Among other things, such standards define how components are uniquely identified, how they are stored on disk, and how they interact with one another and system resources such as the screen.

2.2.2 Storage

Typical container applications keep in persistent storage two versions of the components they embed. The first one consists of the embedded component's native data, which is used to initialize the component. This data is created and managed by the component itself. The second representation is a cached image of the state of the component the last time it was instantiated. This image, although created by the component, is managed by the container application. This image serves two purposes. First, it allows the document to be rendered quickly, since the code that understands the component's specific type need not be executed until the user wishes to modify the component. Second, the cached image allow the document to be rendered even on systems where some components are not available.

2.3 Microsoft Office

In this section we explore the technologies used by MS-Office to enable compound documents. We start with an overview of COM and OLE. We focus on the parts of these technologies that are important for compound document, particularly the Structured Storage Interface. Finally we talk about the two native file formats supported by MS-Office.

2.3.1 COM and OLE

The Component Object Model (COM) enables software components to export well-defined interfaces and interact with one another.

In COM, software components implement their services as one or more COM objects. Every object implement one or more interfaces, each of which exports a number of methods. COM components communicate by invoking these methods.

The Object Linking and Embedding (OLE) specification is a set of standard COM interfaces that enable users to create compound document by *linking* and *embedding* objects (components) into container applications, hence the name OLE. OLE includes other component-based technologies, like ActiveX and OLE Automation.

2.3.2 Structured storage

The OLE Structure Storage Interface (SSI) provides the means for multiple components to share a single file. This is necessary because in the user's perception, a compound document is single unit of information and should appear as a single file.

The SSI implements an abstraction similar to a file system within a single file. It supports types of objects: *storages* and *streams*. Storages are analogous to directories and contains streams or more storages. Stream are analogous to files and contain the components data.

In compound documents, each embedded component is stored in a separate storage. When an embedded components is instantiated, its embedding container supplies it with a pointer to the storage that holds the components native data. The embedded component uses these data to initialize its state.

An embedded component manages its own storage; the parent container need not understand the information stored within it.

2.3.3 Component taxonomy

Conceptually, MS-Office documents may have up to three classes of components: images, OLE-based embedded components, and virtual components. Images are graphic data that are stored and manipulated directly by the application. This includes the cached versions of any embedded components and any graphic data that the application choose to manipulate directly. OLE-based embedded components are data created using a separate application, as described above. Finally, virtual components are objects that are not implemented as OLE-based components but that are perceived by the user as separate entities (*i.e.*, pages in Word, slides in PowerPoint, and sheets in Excel).

2.3.4 File formats

Microsoft Office 2000 supports two native file formats: the traditional OLE-based binary format (hereafter, "OLE archive") and a new XML-based format. The OLE archives [11, 10, 8, 9] rely on the OLE SSI to provide a unified view of the compound document in a single file. However, the manner in which MS-Office applications use the OLE SSI to store embedded object varies. Word and Excel, for example, store every embedded object in a separate storage, making the component structure of the document visible to the OLE SSI. In contrast, PowerPoint compresses embedded object native data and stores it in the main application stream. While this strategy increases document compression, it limits the ability of third-party applications to manipulate components within a PowerPoint document.

The new XML format [12] provides a more browser-friendly option for storing MS-Office documents. Where an OLE archive appears as a single file, an XML document appears as an entire directory of XML files, approximately one per component, image, or slide. The current implementation of MS-Office supports two version of XML output: a compact low-fidelity representation that can be read by browsers but cannot be edited by MS-Office tools, and a larger high quality representation that supports editing. In this study we focus on the latter XML representation.

Aside from the number of files that they use, the two file formats differ mostly in their representation of text and formating information. Images and embedded component native data have similar representation in both formats, with the caveat that component data in the XML-based format is stored in a compressed OLE archive.

3 Data set

We collected Word, Excel, and PowerPoint documents from the Web. First, we used a commercial search engine to obtain an initial set of URLs. We searched for pages having links to files with suffixes we are interested in (`ppt`, `xls`, and `doc`). Then, we used GNU Wget [15] to recursively retrieve documents from our initial search results.

All downloaded documents were in the binary OLE archive format. Because Microsoft's file formats vary from one version of their software to another, we first converted all our data to the Office 2000 formats. We removed documents that appeared to be corrupt or were not actually Office documents; the `doc` suffix, in particular, tends to be used by many applications other than Microsoft Word. We also eliminated duplicates, removing approximately 5% of our data set.

We converted all the data to Office 2000 formats and obtained the XML-based representation, using the MS-Office OLE Automation interfaces [13]. OLE Automation allows a simple Java application we wrote to remotely control the Office applications to perform the data conversions.

Table 1 shows a summary of the documents. For each application, it presents the Internet domains from which documents originated, the number of documents, and the number of sites. The *local* domain corresponds to documents obtained from our local file system. These documents were taken from our local NFS server rather than the Web.

Application	Domain	Documents	Sites
Word	com	412	28
	edu	813	73
	gov	1376	50
	org	362	58
	other	362	27
	local	3007	2
	Subtotal	**6481**	**236**
PowerPoint	com	515	95
	edu	669	73
	gov	333	45
	org	474	111
	other	51	10
	local	125	1
	Subtotal	**2167**	**334**
Excel	com	553	126
	edu	1520	76
	gov	1343	48
	org	448	93
	other	88	35
	local	104	1
	Subtotal	**4056**	**378**
Total		**12704**	**935**

Table 1: Data set. This table presents the domains and sites from which our test documents originated. These numbers reflect the documents that remain after duplicates and corrupted files were removed.

4 Experiments

This section presents statistics we have measured for MS-Office documents and the components within them. We compare our results to previously published statistics for HTML documents.

4.1 Document Size

Table 2 shows general statistics for Word, Excel, and PowerPoint documents[1]. The most striking aspects of the data is the large average size of documents and the large standard deviations of our sample.

Figure 1 shows the size distribution of Word, Excel, and PowerPoint documents. The histogram plots documents with sizes up to 180 KB. We observe that the distributions have the same general shape: a cluster around a common small value with a fairly long tail.

Figure 2 characterizes the distributions' tails by plotting document size frequencies for documents larger than 100 KB on a log-log scale. The linear fit of the transformed data ($y \sim x^{-1.7124}$) with $R^2 = 0.8938$ suggest that the tail of the size distribution follows closely the power-law distribution, which explains the large standard deviations of table 2. The log-log scale histograms for the individual Word, PowerPoint, and Excel documents are not shown here since they are all similar to the cumulative distribution, with linear fits of $y \sim x^{-1.5254}$, $y \sim x^{-1.332}$, $y \sim x^{-1.7485}$, and $R^2 = 0.8612$, $R^2 = 0.8352$, and $R^2 = 0.8226$, respectively.

These results are similar to the findings of Cunha et. al. [5] where the size of HTML-based Web documents was found to follow the power-law distribution. However, while Cunha et. al. found that most HTML documents are quite small (usually between 256 and 512 bytes), MS-Office compound documents tend to be much larger. Common sizes of Word and Excel documents size range from 12 KB to 24 KB, and common PowerPoint documents range from 48 KB to 80 KB. Moreover, the size of compound documents smaller than 40 KB does not follow the power-law distribution.

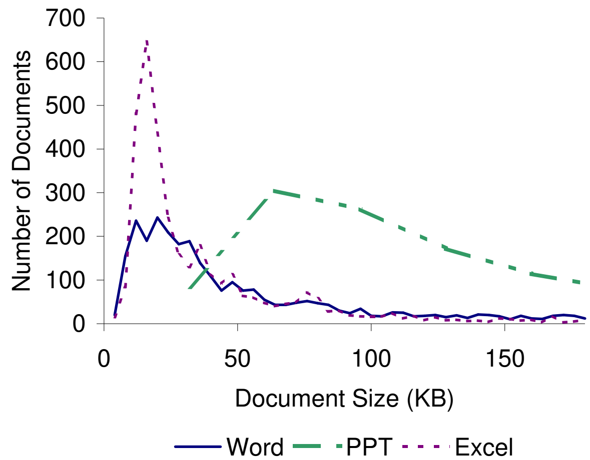

Figure 1: Size distribution of Word, PowerPoint, and Excel documents. Shown are documents with sizes up to 180 KB.

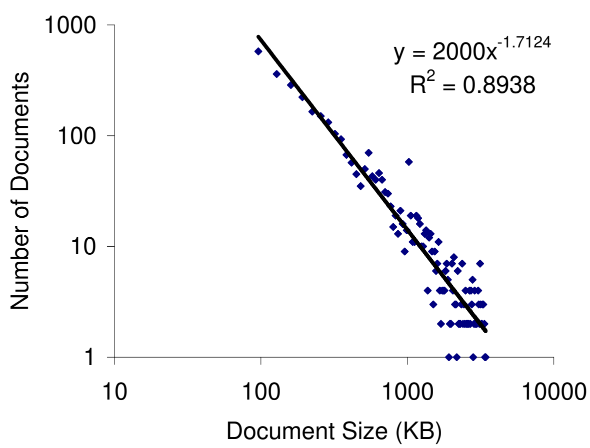

Figure 2: Size distribution of larger MS-Office documents on a log-log scale. Document size frequencies are measured with 16384 byte bins.

4.2 Size breakdown

Figure 3 shows the breakdown of document sizes for Word documents. For every size category it shows the contributions of text, formating information, embedded objects, and images to the documents size.

[1] The document size measurements for table 2 and figures 1 and 2 are based on the *raw* documents retrieved from the Web rather than the normalized Office 2000 translations described in section 3.

Statistic	Application Word	PowerPoint	Excel
average (KB)	196.24	891.48	115.02
stdev (KB)	528.44	2145.35	438.70
median (KB)	47.25	182.25	28.50

Table 2: Document size statistics.

We measured similar breakdowns for PowerPoint and Excel document, however because of space concerns we do not include them in this paper. The PowerPoint documents showed a similar trend to that of Figure 3, while in Excel documents the text component accounts for over 95% of the document size in all the size categories.

Figure 3 show that small Word documents are dominated by text and formating information. However, for larger Word documents, image and embedded component data become the prevalent contributors to document size. This data strongly suggests that efforts to improve access to compound documents should focus on the image and the embedded component data.

One possible optimization would be to remove the embedded component data from documents that are fetched exclusively for reading. As described in section 2.2.2, this data is only necessary when editing an embedded component. Users will still be able to display the document using the cached image of the component. We measured the saving of this schema and found that it would lead to a reduction in bandwidth requirements for Word and PowerPoint documents of up to 35% and 21%, respectively. PowerPoint documents show less potential benefit because PowerPoint compresses its components data before storing it in the OLE archive, whereas Word uses no compression.

4.3 OLE archives vs. XML

In this section we compare documents stored in OLE archives with their XML representations. The results of our comparison are shown in table 3 and figures 4, 5, and 6. The data reveals that the XML representation is significantly larger, requiring up to 5 time more space. XML efficiency is particularly low for small files, which according to our data are the most prevalent. However, XML efficiency improves dramatically as documents get larger.

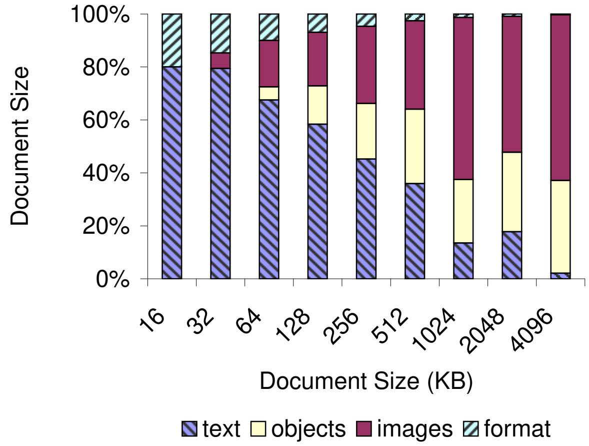

Figure 3: Size breakdown of Word documents. The plot shows that as documents get bigger, images and embedded component data account for most of the document's size.

To understand this, we must understand what happens when a document is converted from an OLE archive to XML. Text and formatting represented in XML takes more space than Microsoft's internal representation. This explains the inefficiency of XML for small files. However, the XML conversion compresses images and embedded component data. PowerPoint already compresses its embedded component data, but Word and Excel do not. Because larger documents tend to be mostly images and components (see figure 3), this explains why the XML representation becomes more efficient for large documents and becomes more efficient than the OLE archive for documents larger than 1 MB.

4.4 Compression

In this section we explore the benefits of compression on the OLE archives and the XML formats. For the OLE archives, we compressed the document by applying gzip to the OLE archive. For the XML for-

Format	Statistic	Application		
		Word	PowerPoint	Excel
OLE	average (KB)	209.19	579.53	110.23
	stdev (KB)	534.59	1671.36	401.83
	median (KB)	55.50	120.25	37.5
gzip OLE	average (KB)	61.43	481.18	25.67
	stdev (KB)	248.89	1597.20	97.88
	median (KB)	12.62	51.62	7.71
XML	average (KB)	226.43	795.17	336.90
	stdev (KB)	583.79	1851.92	1562.04
	median (KB)	70.45	299.15	78.04
gzip XML	average (KB)	74.14	549.03	28.37
	stdev (KB)	297.21	1713.56	92.02
	median (KB)	13.49	106.00	9.37

Table 3: Size statistics for documents in raw OLE, OLE compressed with gzip, raw XML, and XML compressed with gzip. The statistics for OLE differ from those presented in table 2 due to the conversion to MS-Office 2000 formats.

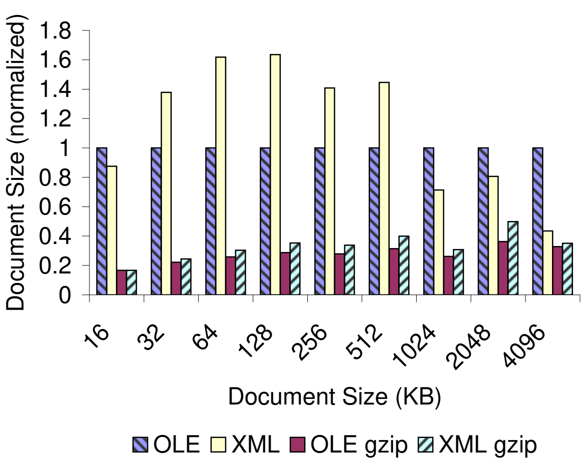

Figure 4: Size distribution of Word documents in OLE archives and XML formats. The plot shows the average sizes of documents in various size categories stored in the OLE archives and the newer XML file format, both with and without compression. All sizes are normalized by the size of the documents stored in the uncompressed OLE archives.

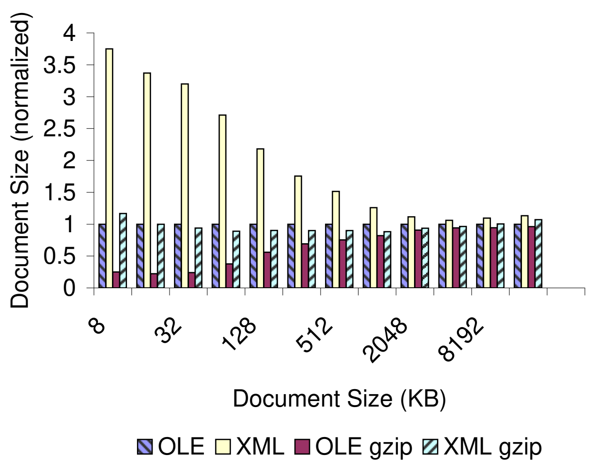

Figure 5: Size distribution of PowerPoint documents in OLE archives and XML formats. The plot shows the average sizes of documents in various size categories stored in the OLE archives and the newer XML file format, both with and without compression. All sizes are normalized by the size of the documents stored in the uncompressed OLE archives.

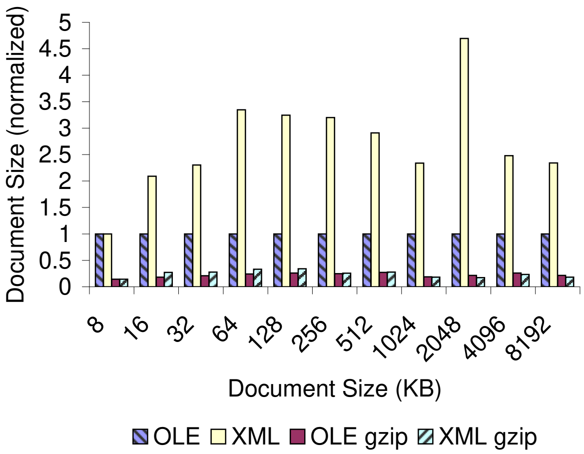

Figure 6: Size distribution of Excel documents in OLE archives and XML formats. The plot shows the average sizes of documents in various size categories stored in the OLE archives and the newer XML file format, both with and without compression. All sizes are normalized by the size of the documents stored in the uncompressed OLE archives.

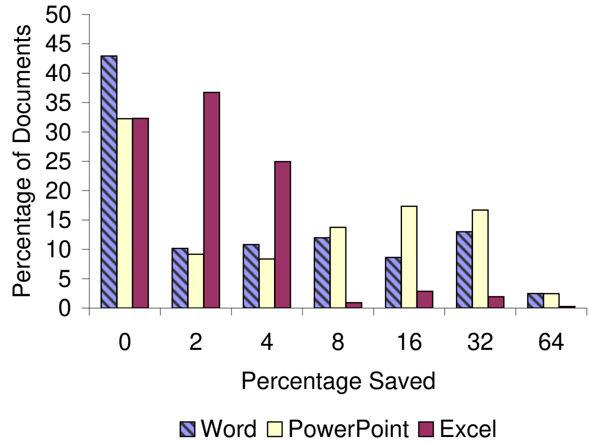

Figure 7: Percentage saved by garbage collection of OLE archive documents.

mat, which uses several files, we compressed each file separately. This strategy emulates the potential benefits of a network infrastructure with built-in compression.

The results of these experiments are shown in table 3 and figures 4, 5, and 6. Compression has a dramatic effect on reducing the size of both OLE archives and XML files; achieving savings as high as 77% for the OLE and 90% for XML. Moreover, the difference in size between compressed OLE and compressed XML representations is small enough to be insignificant. This implies that neither representation has an inherent bandwidth advantage when used across a network.

4.5 Garbage collection

For OLE archives, MS-Office optimizes "save" operations by appending modification to the end of the file rather than rewriting the whole file every time. While this optimization allows for much faster document saves, it can lead to a significant increase in file size. If the user deletes or rewrites a substantial portion of a document and saves it, the original data, now garbage, will still be retained, consuming disk space with no benefit to the user.

When a user asks MS-Office to "save as," a new document is written from scratch, without any garbage that may have been in the original document. We measured the changes in file size for OLE archives by using the "save as" operation. In this experiment we only considered documents that were already in Office 2000 file formats. Other documents are not included because the "save as" operation not only results in garbage collection but also reformats the documents to the Office 2000 formats, potentially increasing or decreasing file size.

Figure 7 shows the results of this experiment. Most documents get some benefit from garbage collection. Impressively, 24% of Word documents and 35% of PowerPoint documents achieve saving greater than 16%.

4.6 Components

In this section we first explore the effects of components on document size. We then present detailed statistics for the three type of components found on MS-Office documents: images, embedded components, and virtual components.

4.6.1 Components and document size

We compared the sizes of MS-Office documents with and without embedded components. Unsurprisingly, documents with embedded components are significantly larger. For example, the average size a Word documents with components is 557.28 KB, relative

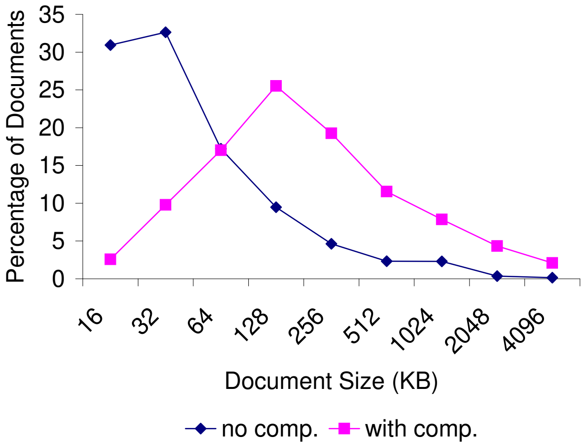

Figure 8: Comparison of size distribution of Word documents with and without components.

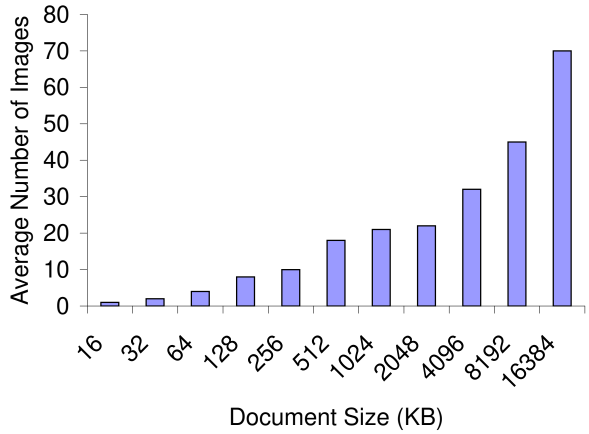

Figure 9: Average number of images in PowerPoint documents.

to an average of 112.32 KB for documents without components. PowerPoint and Excel documents show similar trends: PowerPoint documents average 1334.43 KB with components and 493.58 KB without, and Excel documents average 509.71 KB with components and 109.18 KB without. Further details are presented in figure 8. Despite differences in average file size, it is interesting to note that documents with and without components still follow the same size-distribution shape.

4.6.2 Images

Images are the most common type of non-text data found in MS-Office documents. As table 4 shows, 34.62% of Word and 77.01% of PowerPoint documents on the Web have at least one image. We do not present results for Excel documents as very few of them have any images at all. These results are comparable to the findings of Bray [2], where images were found to be the most common non-text elements in HTML document, and 50% of HTML documents had at least one image.

Figure 9 and 10 show that the average number of images and the average size of images for PowerPoint documents. Both plots show similar trends, with increases in the number and size of images as documents get bigger. These results are consistent with the findings of section 4.2, where the size contribution of images to documents becomes the dominant factor as document size increases. The results for Word are similar, so are omitted for compactness.

We compared the average size of images in MS-Office documents to the findings of previous Web studies [17, 1]. In general these studies report the average size of images between 5 KB and 22 KB. In comparison, MS-Office documents, especially PowerPoint documents, tend to have larger images.

We measured the reuse of images in our PowerPoint documents by calculating the Adler-32 checksum [6] of the image's data and counting the number of documents that have images with the same signatures. We found that from the 16189 images embedded in PowerPoint documents, only 14016 are distinct, while 1241 images or 8.85% appeared in more than one documents. We calculated the potential bandwidth saving of a perfect cache for reading all PowerPoint images and found it to be 15.50%, which correlates well to the proportion of transfers that are saved by the cache - 13.42%. Finally, figure 11 plots the proportion of images that are reused in a given number of document for those images that appear in at least two documents.

4.6.3 Embedded components

The data in table 5 shows that MS-Office documents are rich in component data, with 18.19% of Word document and 46.38% of PowerPoint documents having at least one embedded component. Furthermore, the data shows a high diversity of component types, with Word documents having the highest diversity.

Statistic	Application	
	Word	PowerPoint
% of documents with images	34.62	77.01
average number of images	6.01	10.62
average image size (KB)	21.58	47.82

Table 4: Images statistics for Word and PowerPoint documents. The table shows the percentage of documents that have at least one images, the average number of images in documents that have images, and the average image size.

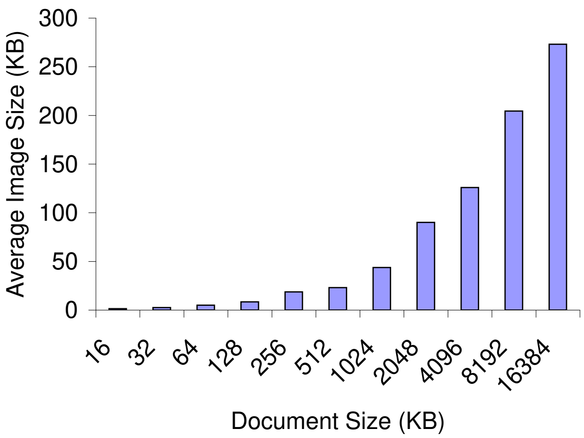

Figure 10: Average image size in PowerPoint documents.

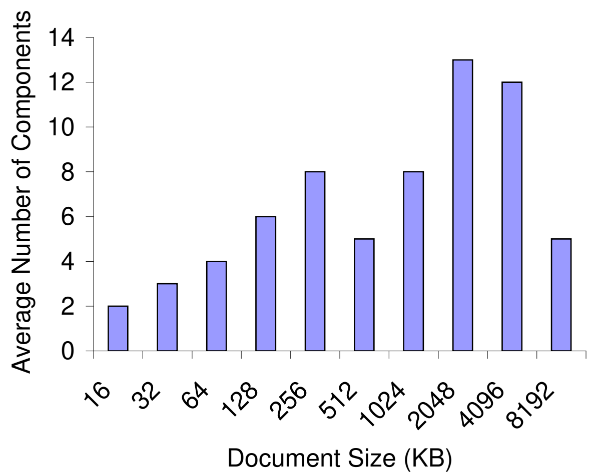

Figure 12: Average number of embedded components in Word documents.

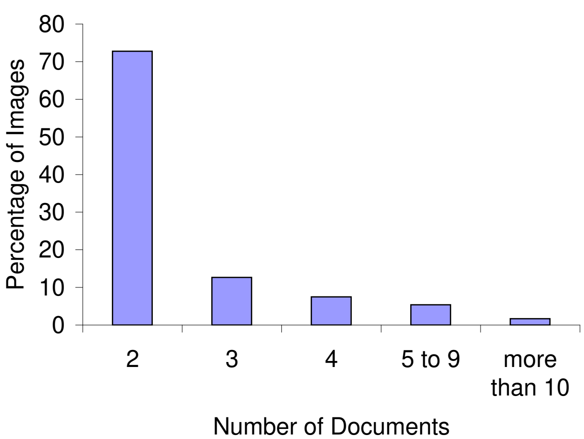

Figure 11: Image reuse. The plot shows the proportion of images that appear in multiple PowerPoint documents.

Table 6 shows the popularity and average size of component types for Word, PowerPoint, and Excel documents. For all three applications, image components are either the first or second most popular type. Additionally, the average size of image components is among the largest of all types. This evidence further suggests that efforts toward reducing file size should focus on image components.

Figure 12 and 13 show that the average number of embedded components and the average size of components for Word documents. As with images, both plots show an increase in the number and size of components as document get bigger. These results are consistent with the findings of section 4.2, where the contribution of embedded components to the document size grows significantly as document size increases. The results for PowerPoint and Excel are similar, so are excluded for compactness.

Statistic	Application Word	PowerPoint	Excel
% with components	18.19	46.38	1.42
number of component types	55	11	8
average number of components	6.71	9.18	9.05
average component size (KB)	37.62	18.51	26.01
stdev (KB)	141.78	109.33	133.37
median (KB)	1.63	2.31	16.81

Table 5: Embedded components statistics. The table shows the percentage of documents that have at least one embedded components, the number of different component types, the average number of components in a document, and the average, standard deviation, and mean of the sizes of embedded components.

Application	Component	Avg. Size (KB)	% of Occurrences
Word	Equation	0.74	51.12
	Word Picture	80.68	14.81
	Clip Art	10.38	8.93
	Excel Sheet	153.46	8.53
	OLE Link	23.80	3.90
	Paint Brush	315.75	1.71
	MS Draw	7.28	1.35
	PowerPoint	41.74	0.96
	Other	97.52	8.68
PowerPoint	Clipart Gallery	5.00	41.12
	Other Image Components	28.68	38.80
	Word Table	23.28	8.01
	Excel	57.13	4.18
	Graph	3.26	3.86
	Equation Editor	0.81	1.82
	Excel Chart	38.08	1.09
	Organization Chart	2.87	0.75
	Note-It OLE	32.11	0.19
	Wordart	1.31	0.15
	Sound	3.35	0.02
Excel	Paint Brush	17.91	0.42
	Word	28.17	0.32
	Clipart Gallery	2.56	0.15
	Forms	2.39	0.05
	Image	4.99	0.05
	Word Picture	2066.83	0.00

Table 6: Average size and popularity of component types in Word, PowerPoint, and Excel documents.

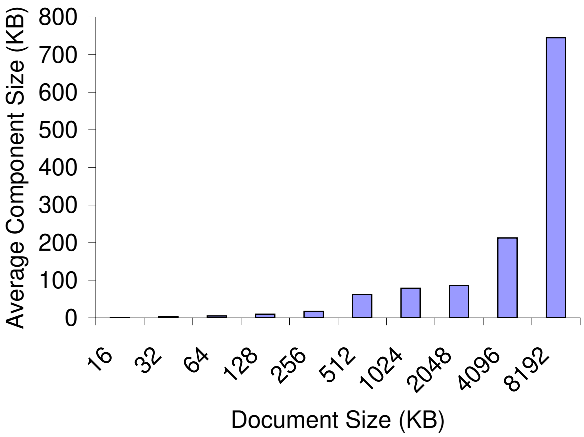

Figure 13: Average size of embedded components in Word documents.

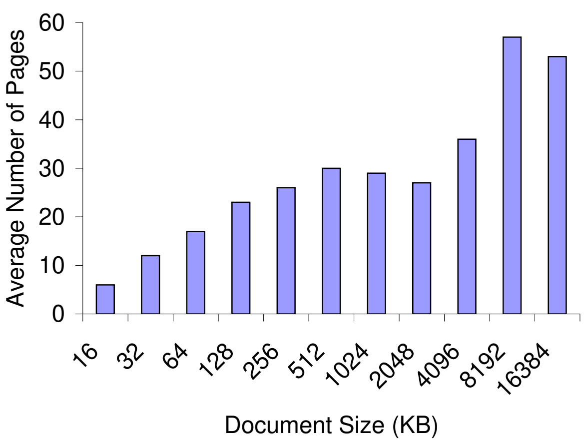

Figure 14: Average number of pages in Word documents.

4.6.4 Virtual components

Table 7 shows the average number of pages, slides, and sheets found in Word, PowerPoint and Excel documents.

Figure 14 shows that the average number of pages increases initially with the size of the document and then levels of. This data would imply that most documents have a similar length and that difference in size are due mainly to the level of sophistication of the document (*i.e.*, whether it has more pictures and embedded components). This is consistent with the findings of section 4.2 where the size of the text element of Word and PowerPoint document remained almost constant over large variations in document size.

5 Conclusions and Discussion

We characterized compound documents collected from the Web and our local file system, generated by the three most popular applications of the Microsoft Office suite: Word, Excel, and PowerPoint. Our study encompasses over 12,500 documents, comprising over 4 GB of data, retrieved from 935 different Web sites.

Our main conclusions are:

1. Compound documents are in general much larger than current HTML documents. The tail of the size distribution follows the same power-log distribution previously observed with HTML documents.

2. For large documents, embedded objects and images comprise the majority of the document.

3. For small documents, XML format produces much larger documents than OLE. For large documents, there is little difference.

4. Compression considerably reduces the size of documents in both XML and OLE archives.

Furthermore, our experience studying the MS-Office file formats resulted in the following insights:

1. The data suggests that the "save as" operation is largely misunderstood by users. The large savings that we show from garbage collection suggest that users do not understand the implications of *fast-save* mode (the default). Moreover, we believe that most user perceive the "save as" operation as just a way to create a copy of the document.

2. The lack of support for compression by the OLE Structured Storage Interface has forced designers to implement ad-hoc solutions in order to achieve high performance. This experience suggests that a compression interface would be a desirable addition to the Structured Storage Interface.

3. OLE archive formats are likely to remain the preferred intermediate format for MS-Office documents, while the XML-based format will

Statistic	Word Pages	PowerPoint Slides	Excel Sheets
average	11.95	20.59	5.22
stdev	27.76	17.48	6.49
median	4	17	2

Table 7: Virtual components. The table shows statistics for pages in Word, slides in PowerPoint, and sheets in Excel documents.

likely be the format of choice for Web publishing. The XML-based format has the advantage that it can potentially be interpreted by application other than MS-Office (*e.g.*, Web browsers). It is also amenable to widespread browser techniques that improve user perceived latency, such as incremental rendering and fetch on-demand. On the flip side, the current implementation of MS-Office 2000 does not implement incremental loading or writing of XML-based documents, leading to higher latencies for opening and storing XML-based documents than those experienced on similar OLE archive documents. Moreover, some of the MS-Office formats do not yet have XML equivalents.

References

[1] ARLITT, M. F., AND WILLIAMSON, C. L. Server workload characterization: the search for invariants. In *ACM SIGMETRICS Conference* (Philadelphia, Pennsylvania, 1996).

[2] BRAY, T. Measuring the Web. In *The World Wide Web Journal* (1996), vol. 1-3.

[3] BROCKSCHMIDT, K. *Inside OLE*. Microsoft Press, 1995.

[4] CHAPPELL, D. *Understanding ActiveX and OLE*. Microsoft Press, 1996.

[5] CUNHA, C. R., BESTAVROS, A., AND CROVELLA, M. E. Characteristics of WWW client-based traces. Tech. Rep. TR-95-010, Boston University, Apr. 1995.

[6] DEUTSCH, P., AND GAILLY, J. L. Zlib compressed data format specification version 3.3. `http://src.doc.ic.ac.uk/Mirrors/ftp.cdrom.com/pub/infozip/doc/rfc1950.txt`, May 1996.

[7] HAMILTON, G. *Java Beans API specification*. Sun Microsystems, July 1997.

[8] MICROSOFT CORPORATION. *Microsoft Excel File Format*. Redmond, Washington, 1997. MSDN Online, `http://msdn.microsoft.com`.

[9] MICROSOFT CORPORATION. *Microsoft Office 97 Drawing File Format*. Redmond, Washington, 1997. MSDN Online, `http://msdn.microsoft.com`.

[10] MICROSOFT CORPORATION. *Microsoft PowerPoint File Format*. Redmond, Washington, 1997. MSDN Online, `http://msdn.microsoft.com`.

[11] MICROSOFT CORPORATION. *Microsoft Word File Format*. Redmond, Washington, 1997. MSDN Online, `http://msdn.microsoft.com`.

[12] MICROSOFT CORPORATION. *Microsoft Office 2000 and HTML*. Redmond, Washington, 1999. MSDN Online, `http://msdn.microsoft.com`.

[13] MICROSOFT PRESS. *Microsoft Office 97 / Visual Basic Programmer's Guide*, 1997.

[14] NELSON, C. Opendoc and its architecture. *AIXpert* (Aug. 1995).

[15] NIKSIC, H. Gnu wget. `http://www.gnu.org/manual/wget/ps/wget.ps`, Sept. 1998.

[16] PITKOW, J. E. Summary of WWW characterizations. In *Proceedings of the Seventh International World Wide Web Conference* (Brisbane, Australia, Apr. 1998).

[17] SEDAYAO, J. "Mosaic will kill my network!" - studying network traffic patterns of Mosaic use. In *Proc. of the 2nd International WWW Conference* (Chicago, Illinois, 1994).

[18] WOODRUFF, A., AOKI, P. M., BREWER, E., GAUTHIER, P., AND ROWE, L. A. An investigation of documents from the World Wide Web. In *The World Wide Web Journal* (1996), vol. 1-3.