APPENDIX F

Linearized PDF

A Linearized PDF file is a file that has been organized in a special way to enable efficient incremental access in a network environment. The file is valid PDF in all respects, and is compatible with all existing viewers and other PDF applications. Enhanced viewer applications can recognize that a PDF file has been linearized and can take advantage of that organization (as well as added hint information) to enhance viewing performance.

The Linearized PDF file organization is an optional feature available beginning in PDF 1.2. Its primary goal is to achieve the following behavior:

- When a document is opened, display the first page as quickly as possible. The first page to be viewed can be an arbitrary page of the document, not necessarily page 0 (though opening at page 0 is most common).
- When the user requests another page of an open document (for example, by going to the next page or by following a link to an arbitrary page), display that page as quickly as possible.
- When data for a page is delivered over a slow channel, display the page incrementally as it arrives. To the extent possible, display the most useful data first.
- Permit user interaction, such as following a link, to be performed even before the entire page has been received and displayed.

This behavior should be achieved for documents of arbitrary size. The total number of pages in the document should have little or no effect on the user-perceived performance of viewing any particular page.

The primary focus of Linearized PDF is optimized viewing of read-only PDF documents. It is intended that the Linearized PDF be generated once and read many times. Incremental update is still permitted, but the resulting PDF is no longer linearized and subsequently is treated as ordinary PDF. Linearizing it again may require reprocessing the entire file; see Section F.4.6, "Accessing an Updated File," for details.

Linearized PDF requires two additions to the PDF specification:

- Rules for the ordering of objects in the PDF file
- Additional data structures, called *hint tables*, that enable efficient navigation within the document

Both of these additions are relatively simple to describe; however, using them effectively requires a deeper understanding of their purpose. Consequently, this appendix goes considerably beyond a simple specification of these PDF extensions to include background, motivation, and strategies.

- Section F.1, "Background and Assumptions," provides background information about the properties of the Web that are relevant to the design of Linearized PDF.
- Section F.2, "Linearized PDF Document Structure," specifies the file format and object-ordering requirements of Linearized PDF.
- Section F.3, "Hint Tables," specifies the detailed representation of the hint tables.
- Section F.4, "Access Strategies," outlines strategies for accessing Linearized PDF over a network, which in turn determine the optimal way to organize the PDF file.

The reader is assumed to be familiar with the basic architecture of the Web, including terms such as URL, HTTP, and MIME.

F.1 Background and Assumptions

The principal problem addressed by the Linearized PDF design is the access of PDF documents through the Web. This environment has the following important properties:

- The access protocol (HTTP) is a transaction consisting of a request and a response. The client presents a request in the form of a URL, and the server sends a response consisting of one or more MIME-tagged data blocks.

- After a transaction has completed, obtaining more data requires a new request-response transaction. The connection between client and server does not ordinarily persist beyond the end of a transaction, although some implementations may attempt to cache the open connection to expedite subsequent transactions with the same server.
- Round-trip delay can be significant. A request-response transaction can take up to several seconds, independent of the amount of data requested.
- The data rate may be limited. A typical bottleneck is a slow modem link between the client and the Internet service provider.

These properties are generally shared by other wide-area network architectures besides the Web. Also, CD-ROMs share some of these properties, since they have relatively slow seek times and limited data rates compared to magnetic media. The remainder of this appendix focuses on the Web.

Some additional properties of the HTTP protocol are relevant to the problem of accessing PDF files efficiently. These properties may not all be shared by other protocols or network environments.

- When a PDF file is initially accessed (such as by following a URL hyperlink from some other document), the file type is not known to the client. Therefore, the client initiates a transaction to retrieve the entire document and then inspects the MIME tag of the response as it arrives. Only at that point is the document known to be PDF. Additionally, with a properly configured server environment, the length of the document becomes known at that time.
- The client can abort a response while the transaction is still in progress if it decides that the remainder of the data is not of immediate interest. In HTTP, aborting the transaction requires closing the connection, which interferes with the strategy of caching the open connection between transactions.
- The client can request retrieval of portions of a document by specifying one or more byte ranges (by offset and count) in the HTTP request headers. Each range can be relative to either the beginning or the end of the file. The client can specify as many ranges as it wants in the request, and the response consists of multiple blocks, each properly tagged.
- The client can initiate multiple concurrent transactions in an attempt to obtain multiple responses in parallel. This is commonly done, for instance, to retrieve inline images referenced from an HTML document. This strategy is not

always reliable and may backfire if the transactions interfere with each other by competing for scarce resources in the server or the communication channel.

Note: *Extensive experimentation has determined that having multiple concurrent transactions does not work very well for PDF in some important environments. Therefore, Linearized PDF is designed to enable good performance to be achieved using only one transaction at a time. In particular, this means that the client must have sufficient information to determine the byte ranges for all the objects required to display a given page of the PDF file so that it can specify all those byte ranges in a single request.*

The following additional assumptions are made about the PDF viewer application and its local environment:

- The viewer application has plenty of local temporary storage available. It should rarely need to retrieve a given portion of a PDF document more than once from the server.
- The viewer application is able to display PDF data quickly once it has been received. The performance bottleneck is assumed to be in the transport system (throughput or round-trip delay), not in the processing of data after it arrives.

The consequence of these assumptions is that it may be advantageous for the client to do considerable extra work to minimize delays due to communications. Such work includes maintaining local caches and reordering actions according to when the needed data becomes available.

F.2 Linearized PDF Document Structure

Except as noted below, all elements of a Linearized PDF file are as specified in Section 3.4, "File Structure," and all indirect objects in the file are numbered sequentially in two groups, based on their order of appearance in the file.

- The first group consists of the document catalog, certain other document-level objects, and all objects belonging to the first page of the document. These objects are numbered sequentially, starting at the first object number after the last number of the second group. (The stream containing the hint tables, called a *hint stream*, may be numbered out of sequence; see Section F.2.5, "Hint Streams (Parts 5 and 10).")

- The second group consists of all remaining objects in the document, including all pages after the first, all shared objects (objects referenced from more than one page, not counting objects referenced from the first page), and so forth. These objects are numbered sequentially starting at 1.

These groups of objects are indexed by exactly two cross-reference table sections, located as shown in Example F.1. The composition of these groups is discussed in more detail in the sections that follow (ordered by the part number as shown in this example, with one section for parts 5 and 10). All objects have a generation number of 0.

Beginning with PDF 1.5, PDF files may contain object streams (see Section 3.4.6, "Object Streams"). In linearized files containing object streams, the following conditions apply:

- Certain additional objects cannot be contained in an object stream: the linearization dictionary, the document catalog, and page objects.
- Objects stored within object streams are given the highest range of object numbers within the main and first-page cross-reference sections.
- For files containing object streams, hint data can specify the location and size of the object streams only (or uncompressed objects), not the individual compressed objects. Similarly, shared object references should be made to the object stream containing a compressed object, not to the compressed object itself.
- Cross-reference streams (Section 3.4.7, "Cross-Reference Streams") can be used in place of traditional cross-reference tables. The logic described in this chapter still applies, with the appropriate syntactic changes.

Example F.1

Part 1: Header

```
%PDF–1.1          % …Binary characters…
```

Part 2: Linearization parameter dictionary

```
43 0 obj
    << /Linearized 1.0          % Version
        /L 54567                % File length
        /H [475 598]            % Primary hint stream offset and length (part 5)
        /O 45                   % Object number of first page's page object (part 6)
        /E 5437                 % Offset of end of first page
        /N 11                   % Number of pages in document
        /T 52786                % Offset of first entry in main cross-reference table (part 11)
    >>
endobj
```

Part 3: First-page cross-reference table and trailer

```
xref
43 14
0000000052 00000 n
0000000392 00000 n
0000001073 00000 n
…Cross-reference entries for remaining objects in the first page…
0000000475 00000 n
trailer
    << /Size 57                 % Total number of cross-reference table entries in document
        /Prev 52776             % Offset of main cross-reference table (part 11)
        /Root 44 0 R            % Indirect reference to catalog (part 4)
        …Any other entries, such as Info and Encrypt…         % (part 9)
    >>
startxref
0                               % Dummy cross-reference table offset
%%EOF
```

Part 4: Document catalog and other required document-level objects

```
44 0 obj
    << /Type /Catalog
        /Pages 42 0 R
    >>
endobj

…Other objects…
```

Part 5: Primary hint stream (may precede or follow part 6)

```
56 0 obj
    << /Length 457
        …Possibly other stream attributes, such as Filter…
        /S 221                      % Position of shared object hint table
        …Possibly entries for other hint tables…
    >>
stream
    …Page offset hint table…
    …Shared object hint table…
    …Possibly other hint tables…
endstream
endobj
```

Part 6: First-page section (may precede or follow part 5)

```
45 0 obj
    << /Type /Page
        …
    >>
endobj
```

…Outline hierarchy (if the PageMode value in the document catalog is UseOutlines)…

…Objects for first page, including both shared and nonshared objects…

Part 7: Remaining pages

```
1 0 obj
    << /Type /Page
        …Other page attributes, such as MediaBox, Parent, and Contents…
    >>
endobj
```

…Nonshared objects for this page…

…Each successive page followed by its nonshared objects…

…Last page followed by its nonshared objects…

Part 8: Shared objects for all pages except the first

…Shared objects…

Part 9: Objects not associated with pages, if any

…Other objects…

Part 10: Overflow hint stream (optional)

…Overflow hint stream…

Part 11: Main cross-reference table and trailer

```
xref
0  43
0000000000  65535  f
…Cross-reference entries for all except first page's objects…
trailer
    << /Size  43 >>              % Trailer need not contain other entries; in particular,
startxref                        %     it should not have a Prev entry
257                              % Offset of first-page cross-reference table (part 3)
%%EOF
```

F.2.1 Header (Part 1)

The Linearized PDF file begins with the standard header line (see Section 3.4.1, "File Header"). Linearization is independent of PDF version number and can be applied to any PDF file of version 1.1 or greater.

The binary characters following the percent sign on the second line are characters with codes 128 or greater, as recommended in Section 3.4.1, "File Header."

F.2.2 Linearization Parameter Dictionary (Part 2)

Following the header, the first object in the body of the file (part 2) must be an indirect dictionary object, the *linearization parameter dictionary*, containing the parameters listed in Table F.1. All values in this dictionary must be direct objects. There are no references to this dictionary anywhere in the document; however, the first-page cross-reference table (Part 3) contains a normal entry for it.

The linearization parameter dictionary must be entirely contained within the first 1024 bytes of the PDF file. This limits the amount of data a viewer application must read before deciding whether the file is linearized.

TABLE F.1 Entries in the linearization parameter dictionary

PARAMETER	TYPE	VALUE
Linearized	number	*(Required)* A version identification for the linearized format. As usual, a change in the integer part indicates an incompatible change in the linearized format, while a change in the fractional part indicates a backward-compatible change. The current version is 1.0.
L	integer	*(Required)* The length of the entire file in bytes. It must be exactly equal to the actual length of the PDF file. A mismatch indicates that the file is not linearized and must be treated as ordinary PDF, ignoring linearization information. (If the mismatch resulted from appending an update, the linearization information may still be correct but requires validation; see Section F.4.6, "Accessing an Updated File," for details.)
H	array	*(Required)* An array of two or four integers, [$offset_1$ $length_1$] or [$offset_1$ $length_1$ $offset_2$ $length_2$]. $offset_1$ is the offset of the primary hint stream from the beginning of the file. (This is the beginning of the stream object, not the beginning of the stream data.) $length_1$ is the length of this stream, including stream object overhead. If the value of the primary hint stream dictionary's **Length** entry is an indirect reference, the object it refers to must immediately follow the stream object, and $length_1$ also includes the length of the indirect length object, including object overhead. (See implementation note 178 in Appendix H.) If there is an overflow hint stream, $offset_2$ and $length_2$ specify its offset and length. (See implementation note 179 in Appendix H.)
O	integer	*(Required)* The object number of the first page's page object.
E	integer	*(Required)* The offset of the end of the first page (the end of part 6 in Example F.1), relative to the beginning of the file. (See implementation note 180 in Appendix H.)
N	integer	*(Required)* The number of pages in the document.

PARAMETER	TYPE	VALUE
T	integer	*(Required)* In documents that use standard main cross-reference tables (including hybrid-reference files; see *"Compatibility with Applications That Do Not Support PDF 1.5" on page 109*), this entry represents the offset of the white-space character preceding the first entry of the main cross-reference table (the entry for object number 0), relative to the beginning of the file. Note that this differs from the **Prev** entry in the first-page trailer, which gives the location of the **xref** line that precedes the table. In PDF 1.5 and later documents that use cross-reference streams exclusively (see Section 3.4.7, "Cross-Reference Streams"), this entry represents the offset of the main cross-reference stream object.
P	integer	*(Optional)* The page number of the first page (see Section F.2.6, "First-Page Section (Part 6)"). Default value: 0.

F.2.3 First-Page Cross-Reference Table and Trailer (Part 3)

Part 3 contains the cross-reference table for objects belonging to the first page (discussed in Section F.2.6, "First-Page Section (Part 6)") as well as for the document catalog and document-level objects appearing before the first page (discussed in Section F.2.4, "Document Catalog and Document-Level Objects (Part 4)"). Additionally, this cross-reference table contains entries for the linearization parameter dictionary (at the beginning) and the primary hint stream (at the end). This table is a valid cross-reference table as defined in Section 3.4.3, "Cross-Reference Table," although its position in the file is unconventional. It consists of a single cross-reference subsection that has no free entries.

Note: *In PDF 1.5 and later, cross-reference streams (see Section 3.4.7, "Cross-Reference Streams") may be used in linearized files in place of traditional cross-reference tables. The logic described in this section, along with the appropriate syntactic changes for cross-reference streams, still applies.*

Below the table is the first-page trailer. The trailer's **Prev** entry gives the offset of the main cross-reference table near the end of the file. This is valid PDF syntax, although the trailers are linked in an unusual order. A PDF viewer application that is unaware of linearization interprets the first-page cross-reference table as an update to an original document that is indexed by the main cross-reference table.

The first-page trailer must contain valid **Size** and **Root** entries, as well as any other entries needed to display the document. The **Size** value must be the combined number of entries in both the first-page cross-reference table and the main cross-reference table.

The first-page trailer may optionally end with **startxref**, an integer, and %%EOF, just as in an ordinary trailer. This information is ignored.

F.2.4 Document Catalog and Document-Level Objects (Part 4)

Following the first-page cross-reference table and trailer are the catalog dictionary and other objects that are required when the document is opened. These additional objects (constituting part 4) include the values of the following entries if they are present and are indirect objects:

- The **ViewerPreferences** entry in the catalog.
- The **PageMode** entry in the catalog. Note that if the value of **PageMode** is UseOutlines, the outline hierarchy is located in part 6; otherwise, the outline hierarchy, if any, is located in part 9. See Section F.2.9, "Other Objects (Part 9)" for details.
- The **Threads** entry in the catalog, along with all thread dictionaries it refers to. This does not include the threads' information dictionaries or the individual bead dictionaries belonging to the threads.
- The **OpenAction** entry in the catalog.
- The **AcroForm** entry in the catalog. Only the top-level interactive form dictionary is needed, not the objects that it refers to.
- The **Encrypt** entry in the first-page trailer dictionary. All values in the encryption dictionary must also be located here.

Objects that are not ordinarily needed when the document is opened should not be located here but instead should be at the end of the file; see Section F.2.9, "Other Objects (Part 9)." This includes objects such as page tree nodes, the document information dictionary, and the definitions for named destinations.

Note that the objects located here are indexed by the first-page cross-reference table, even though they are not logically part of the first page.