A Generalized Grouping and Retrieval Scheme for Stored MPEG Video*

Senthil Sengodan, Victor O. K. Li

Communication Sciences Institute
Department of Electrical Engineering - Systems
University of Southern California
Los Angeles, CA 90089-2565, U.S.A.
E-mail: {sengodan,vli}@milly.usc.edu

Abstract

MPEG, in addition to being an international standard, is currently the most popular coding scheme for stored video. For several applications that require stored video, such as video-on-demand, it is advantageous to group MPEG frames into segments. Retrieving segments instead of individual frames can result in a more efficient use of disk retrieval resources and consequently, in the support of a larger number of users and in a more cost-effective system. Depending on the value of parameters such as desired user quality of service, size and cost of required buffers, number of disks to stripe a segment across, etc., a certain segment length (defined as the number of frames per segment) may be preferred over others. In this paper, we propose a generalized grouping scheme that may be used to achieve any desired segment length. Properties of such a grouping scheme and formulae that may be used to achieve such a grouping have been provided. Since a segment is an atomic unit of video retrieval, retrieval of some segments is skipped during fast playback. Some frames belonging to retrieved segments may be discarded due to the unavailability of other frames necessary for their decoding. Tradeoffs between different kinds of fast playback (i.e., the choice of skipped segments) and the number of discarded frames is discussed.

1 Introduction

1.1 MPEG coding

Efficient storage and retrieval of video is extremely important for the success of emerging applications such as Video-on-demand (VOD) [2, 3, 4, 7, 6]. The most popular method of coding for applications that require stored video is the international standard Motion Pictures Expert Group (MPEG) coding [5]. An MPEG coded video contains three different kinds of frames - Intraframe (I), Predictive (P) and Bidirectional (B). An I frame may be decoded independently; a P frame requires the previous I/P frame for decoding; and, a B frame requires the previous as well as the subsequent I/P frame for decoding. For instance, in Figure 2(a), frame 3 is needed to decode frame 6 while frames 6 and 9 are needed to decode frames 7 as well as 8. The ratio of the number of I, P and B frames is a coding parameter and may be varied depending on the desired video quality and storage size. We will denote the number of P frames between two consecutive I frames by $(m-1)$, and the number of B frames between two consecutive I/P frames by $(q-1)$. Each frame is assigned a frame number which depends on its position in the playback order. The pattern indicated in Figure 2(a) repeats, i.e., frame number 21 is an I frame, frame number 22 is a B frame, and so on.

1.2 Video segment

A video segment is a group of frames and is considered to be an atomic (indivisible) unit for retrieval purposes. Since the average size of a segment is larger than that of a frame, a segment is better suited for striping across a larger number of disks. This is because disk data is stored in blocks and striping one frame across several disks may result in a storage of less than one block per disk. This means that during retrieval the entire block is retrieved whereas only a small portion of the retrieved data is useful. Striping across a larger number of disks has the advantage that a larger retrieval bandwidth may be obtained. Since a disk head has to seek the block in which the data to be retrieved is stored and since different blocks storing portions of the same segment are usually located on the same disk track, retrieval of a larger amount of data (i.e., a segment instead of a frame) results in a decrease in the average seek time for a unit of video data. Due to these reasons, frames are often grouped into segments and a segment is taken to be the smallest (atomic) unit of video retrieval.

While a larger segment size has the advantage of accommodating a larger number of users in the system because of a more efficient use of disk retrieval resources, it also suffers from certain drawbacks. A large segment size requires more buffer space (at the server and/or at the user's set-top box) which adds to the cost of the system. Hence, a tradeoff exists between the number of users that can be supported and the required buffer. Based on the relative weights of these two parameters, a certain segment length (defined as the number of frames per segment) may be preferred. Hence, suitable and efficient grouping of frames for this segment length is needed.

1.3 Frame/segment skipping

During fast playback, a desirable and often used approach is to skip certain frames and display others at the

*This research is supported in part by the TRW Foundation and in part by the Pacific Bell External Technology Program.

0-7803-3925-8/97 $10.00 ©1997 IEEE

Frame Number : 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

Fast Playback : $n_1 = 2n$

Piece Size	Displayed Frames
1	0 2 4 6 8 10 12 14 16 18
2	0 1 4 5 8 9 12 13 16 17
5	0 1 2 3 4 10 11 12 13 14

Figure 1: Piecewise cont. display for fast playback

same rate as that of normal playback. This is because the frame display rate (or refresh rate) of video monitors as well as TV screens is constant, i.e., frames can only be displayed at a fixed rate. Let us denote the normal playback rate by n frames per second (fps) and the fast playback rate by n_1 fps where $n_1 > n$. This fast playback is simulated by displaying n out of every n_1 frames in the playback order at the same rate of frame display as normal playback. The choice of frames to be displayed determines the nature and quality of fast playback display.

We shall refer to a fast playback as being a *piecewise continuous* display of consecutive frames at the normal playback rate. Each *piece* in this piecewise continuous display comprises consecutive frames in the playback order. During the period that a piece is played back, the portion of the video that comprises the piece is being viewed at the normal playback rate. Hence, in order to display n out of n_1 frames, the piecewise continuous playback may either comprise several short pieces or a few long pieces. In the case of the former, the size of the break between pieces (i.e., the number of consecutive frames not displayed) is short while in the latter it is long. Depending on the desired fast playback appearance, the piece size is determined. This is illustrated in Figure 1 where the fast playback rate (n_1) is twice the normal playback rate (n). The frames that would be displayed for three piece sizes (1, 2 and 5) is indicated.

We just saw that some frames are not displayed during fast playback. In order to decrease the load on the video server as well as the network (i.e., in order to conserve disk retrieval as well as network resources), it is desirable that these frames not be retrieved from the server and transported to the user via the network. Since the atomic unit of retrieval is a video segment, this implies that retrieval of only certain segments takes place from the video server, i.e., retrieval of certain segments is skipped. It is also desirable that all frames belonging to retrieved segments be displayed. However, this may not be possible because decoding of a frame may require other frames that were not retrieved. Consequently, grouping of frames into segments and the choice of segments to be retrieved for fast playback should be such that the number of retrieved frames that are not displayed is minimized.

1.4 Research motivation

Although the advantages of grouping frames into segments has been realized by several researchers, not much work has been done on how such a grouping can be done. In [1], a segment constitutes all frames from one I frame to (and excluding) the next. The number of frames per

(a) Playback Order : I B B P B B P B B P B B P B B P B B P B B
Frame Number (i) : 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

(b) Minimal Causal Order : I P B B P B B P B B P B B P B B P B B I B B
Position (s_i) : 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
Frame Number (i) : 0 3 1 2 6 4 5 9 7 8 12 10 11 15 13 14 18 16 17 21 19 20

(c) Minimal Causal Order: I B B P B B P B B P B B P B B P B B P B B I B B
(with dummy frames)
Position (s_i) : 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
Frame Number (i) : 0 x x 3 1 2 6 4 5 9 7 8 12 10 11 15 13 14 18 16 17 21 19 20

Figure 2: Minimal causal order

segment, here, is qm and fixed by the coding parameters q and m. In this paper, we propose a generalized scheme for grouping frames into segments of any desired segment length. Closed form equations are provided that may be used to easily determine the frames to be grouped together into a segment. Guidelines are provided to determine the segments to be retrieved for any playback rate. Tradeoffs between different kinds of fast playback (i.e., different piece sizes) and the number of discarded frames is discussed.

2 Grouping scheme

2.1 Minimal causal order

The playback order, indicated in Figure 2(a), is not causal with respect to frame requirement for frame decoding. A frame in the playback order may require frames later in the order for its decoding. Consider the case where frames are retrieved periodically and one at a time according to the playback order. The time period within which one frame is retrieved equals the time for playing back one frame. From Table 1(a), it is seen that q frames need to be prefetched prior to the start of playback in order to avoid video starvation. The maximum buffer occupancy is q frames.

Since only the B frames require frames after it in the playback order for decoding purposes, moving every I/P frame in the playback order $(q-1)$ places up would result in a causal order. This causal order is shown in Figure 2 and is referred to as the *minimal causal order*. Let frames be retrieved periodically and one at a time according to the minimal causal order, with the time period of frame retrieval being equal to the frame playback duration. As shown in Table 1(b), only two frames have to be prefetched prior to start of playback in order to avoid video starvation. The maximum buffer occupancy is two frames.

Theorem 1: Let frames be retrieved one at a time and periodically with the retrieval period equaling the playback period. The number of frames that need to be prefetched prior to start of playback (in order to avoid video starvation) and consequently the maximum number of frames that need to be buffered is minimized when the order of frame retrieval follows the minimal causal order.

Proof: When frames are retrieved according to the minimal causal order, irrespective of the values of q and m, only two frames need to be prefetched prior to start of playback. The maximum number of frames that need to be buffered is

Time	0	1	2	3	4	5	6	7	8	9	10
(a) Playback order retrieval											
Retrieved frame	0	1	2	3	4	5	6	7	8	9	10
Displayed frame	-	-	0	1	2	3	4	5	6	7	8
# frames in buffer	1	2	3	3	3	3	3	3	3	3	3
(b) Minimal causal order retrieval											
Retrieved frame	0	3	1	2	6	4	5	9	7	8	12
Displayed frame	-	0	1	2	3	4	5	6	7	8	9
# frames in buffer	1	2	2	2	2	2	2	2	2	2	2

Table 1: Playback/minimal causal order retrieval

two. Prefetching only one frame would not work for any order of frame retrieval because of the decoding requirements of B frames. When a B frame is to be displayed, the I/P frame following it in the playback order needs to be present in the buffer for the successful decoding of the B frame. This means that we need at least a buffer storage of two frames. □

Storage and retrieval of MPEG frames according to a causal order results in a simpler decoding of frames. The earlier that a certain frame appears in the causal order compared to its position in the playback order, the longer it needs to be buffered. A frame whose position in the causal order is later than its position in the playback order does not have to be buffered. If $i = \{0, 1, \cdots\}$ denotes the frame number and $s_i = \{0, 1, \cdots\}$ the corresponding position of frame i in a causal order, then

$$max(i - s_i + p - 1)^+, \qquad \forall\, i = \{0, 1, \cdots\}$$

is an indicator of the maximum time between any frame retrieval and playback.

Here, p denotes the number of frames that have to be prefetched and

$$(n)^+ = \begin{cases} n, & if\ n \geq 0, \\ 0 & if\ n < 0. \end{cases}$$

Theorem 2: Of all possible causal orders, the minimal causal order minimizes $max(i - s_i + p - 1)^+, \forall\, i = \{0, 1, \cdots\}$.
Proof: The requirement of the I/P frame following a B frame for the decoding of the B frame is what makes the playback order non-causal. Moving each I/P frame in the playback order $(q-1)$ places ahead would result in a causal order which is the minimal causal order. Since the minimal causal order was obtained by the least modifications to the playback order, it minimizes $max(i - s_i)^+$. Moreover, the minimal causal order has $p = 2$ which is the smallest of all causal orders. Hence, the result. □

From Figure 2(b), it is seen that the first I and the first P frames appear consecutively in the minimal causal order. Any subsequent I/P frame is separated from its neighboring I/P frame by $(q-1)$ B frames in the storage order. In order to make the start of the storage order similar to the rest, we introduce $(q-1)$ dummy B frames between the first I and P frames. The frame numbers of these frames have been denoted by x and the resulting order is shown in Figure 2(c). Similarly, any boundary effects at the end of the video may also be resolved by introducing suitable dummy frames. Henceforth, we shall only deal with the causal order that includes the dummy frames and refer to such a causal order as the minimal causal order (instead of the one without the dummy frames).

2.2 Grouping

We shall define a *frame storage order* to be an arrangement of frames using which the frames that belong to a particular segment can be directly determined. Let the segment length be s frames per segment. In order to group frames into segments of length s, every set of s consecutive frames beginning at the start of the frame storage order are grouped together. Due to the properties enjoyed by the minimal causal order, we shall choose the minimal causal order as the frame storage order. Just like frames, each segment is assigned a *segment number* with the first segment being assigned a number zero. We shall refer to the arrangement of segments in ascending segment numbers as the *segment storage order*. It is easy to see that the segment storage order is causal with regard to frame decoding whenever the frame storage order is causal. When the frame storage order is causal, a frame belonging to any segment does not require a frame belonging to a segment later in the order (i.e., a segment with a larger segment number) for decoding purposes. We define a *segment set* to be a set comprising the smallest number (say, c) of consecutive segments in the storage order that form a segment pattern. The first segment set includes c consecutive segments beginning with the first segment of the segment storage order, i.e., segment number 0.

Property 1: The number of segments in a segment set equals $c = \frac{LCM(qm,s)}{s}$.
Proof: The minimum number of frames in the frame storage order that form a frame pattern is qm. Since sets of s consecutive frames in the frame storage order constitute a segment, the number of frames encountered before a segment pattern repeats is $LCM(qm, s)$. Thus, the number of segments encountered before a segment pattern repeats is $\frac{LCM(qm,s)}{s}$ segments. □

Thus, segments with numbers $(cj + i)$ are similar segments for all $i = \{0, 1, \cdots, c-1\}$ and all nonnegative j, i.e., they contain the same kinds of frames (I, P or B) in the same order. In Figure 3, for the values $q = 3$, $m = 7$, $s = 2$, we have $c = \frac{LCM(21,2)}{2} = 21$. Segments with numbers 1, 22, 43, $\cdots$ are similar segments, which is a BP segment in this case.

Property 2: The s frames contained in a segment with number i have the positions $is, is+1, is+2, \cdots, is+s-1$ in the frame storage order.
Proof: This immediately follows from the fact that every set of s consecutive frames beginning at the start of the frame storage order constitute a segment. □

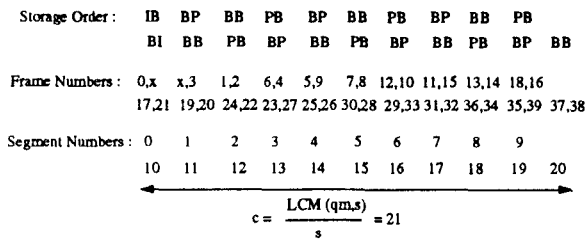

Figure 3: Grouping scheme

Property 3: A frame with position j in the storage order has a frame number (which is its position in the playback order) of $j - \lceil \frac{j}{q} - \lfloor \frac{j}{q} \rfloor \rceil q$.

Proof: The position of an I/P frame in the storage order is the same as that in the playback order (i.e., equals its frame number). This means that if the frame with position j in the storage order is an I/P frame, then it's frame number should equal j. A frame is an I/P frame if and only if its position j in the storage order is a multiple of q. In such a case, the expression above equals j since the second term goes to zero.

The position of a B frame in the playback order is $(q-1)$ places ahead of its position in the frame storage order. This means that if the frame with position j in the storage order is a B frame, then it's frame number should be $(j-q)$. A frame is a B frame if and only if its position j in the storage order is not a multiple of q. The term $\lceil \frac{j}{q} - \lfloor \frac{j}{q} \rfloor \rceil$ equals one whenever j is not a multiple of q, and the expression above equals $(j-q)$. The non-existent B frames (denoted by x) return a negative frame number (because they are the frames before the I frame with number 0) and can be easily identified. □

Using the expressions in Properties 2 and 3, given any segment number (i.e., the position of any segment in the segment storage order), we can determine the frame number of each frame in the segment. With a knowledge of the frame number of each frame contained in each segment of any segment set, the frame numbers of the frames contained in any other segment may be easily determined. This is because corresponding segments in consecutive segment sets contain frames whose frame numbers are offset by cs.

3 Fast playback and segment skipping

We shall now discuss segment retrieval during fast playback. We are interested in two parameters - the piece length and the percentage of retrieved frames that are discarded.

3.1 Grouping properties useful for segment retrieval

The normal playback rate is n fps and the fast playback rate is n_1 fps where $n_1 > n$. On an average, n out of every n_1 frames need to be displayed. Canceling the highest common factor (HCF) between the two, we need to display $\frac{n}{HCF(n,n_1)}$ out of every $\frac{n_1}{HCF(n,n_1)}$ frames. The atomic unit of retrieval is not a frame but a segment. If we know the segments to be retrieved out of the segments belonging to an integral number (say, d) of consecutive segment sets, then the segments to be retrieved in any segment set can be immediately determined. This is because the segments to be retrieved in segment set j are similar to those retrieved in segment set $(j \ mod \ d)$. The number d is the smallest integer such that the ratio of the number of frames displayed to the number of frames in d segment sets equals n/n_1. Since a segment set contains sc frames, d is chosen such that the number of frames in d segment sets equals $LCM(sc, \frac{n_1}{HCF(n,n_1)})$. Hence, we need to display $LCM(sc, \frac{n_1}{HCF(n,n_1)}) \times \frac{n}{n_1}$ out of every $LCM(sc, \frac{n_1}{HCF(n,n_1)})$ frames. We will denote the former number by a and the latter by b. We need to display a out of b frames. We have $d = b/(sc)$. In Figure 4, d rows of segments are shown with each row containing segments belonging to the same segment set.

In this paper, we will assume that a segment contains at most one I frame. This means that the maximum segment size s equals qm. Extension to the case of a larger segment size can be similarly done and is not dealt with here.

Property 4: Out of the c segments in a segment set, the number of segments that contain an I frame equals $\frac{cs}{qm}$.

Proof: The number of frames in a segment set is cs, and from the definition of a segment set, this number is a multiple of qm. Since each set of qm frames contains exactly one I frame, the number of segments containing an I frame equals $\frac{cs}{qm}$. □

Property 5: (a) Of all segments that contain an I frame in them, the l-th segment (in the segment storage order) can be represented as $l = i \times \frac{cs}{qm} + j$ for integers $l, i = \{0, 1, 2, \cdots\}$ and $j = \{0, 1, \cdots, \frac{cs}{qm} - 1\}$.

(b) The segment number of the l-th such segment is $ic + \lfloor j \times \frac{qm}{s} \rfloor$.

(c) The position of the I frame in the l-th such segment is $jqm - s\lfloor j \times \frac{qm}{s} \rfloor$.

Proof: (a) This follows directly from Property 4. i denotes the segment set that the l-th segment with an I frame in it belongs to, while j is the number of segments in segment set i that are ahead of the concerned segment (in the segment storage order) and that have an I frame in them.

(b) Since i denotes the segment set containing the l-th segment with an I frame in it, the number of segments in previous segment sets equals ic. $\lfloor j \times \frac{qm}{s} \rfloor$ denotes the number of segments in segment set i that are ahead of the concerned segment (in the segment storage order).

(c) Since a segment set contains a set of segments that form a segment pattern in the segment storage order, the position of the I frame in the l-th segment with an I frame in it does not depend on the value of i. jqm denotes the number of frames in segment set i that are ahead of the I frame of the concerned segment (in the frame storage order). $s\lfloor j \times \frac{qm}{s} \rfloor$ denotes the number of frames in segment set i that are ahead of the I frame of the concerned segment (in the frame storage order) and which do not belong to the concerned segment. The difference of these two terms gives the number of frames ahead of the I frame (in the frame storage order) in the concerned segment. This is merely the position of the I frame in the concerned segment. □

Property 6: When a segment with an I frame is retrieved while the segment before it in the segment storage order

is not, the number of frames in the retrieved segment that have to be discarded equals $p_I + \min(s-p_I-1, q-1)$. Here, p_I denotes the position of the I frame within the segment.

Proof: When a segment with an I frame in it is retrieved while the segment before it in the segment storage order is not, all frames belonging to the retrieved segment and that are ahead of the I frame (in the frame storage order) need to be discarded. The number of such frames equals p_I. In addition, the $(q-1)$ B frames that follow the I frame (in the frame storage order) also cannot be decoded and have to be discarded. The number of such B frames belonging to the retrieved segment equals $\min(s-p_I-1, q-1)$. □

Property 7: When segments are retrieved consecutively (in the segment storage order) starting with a segment containing an I frame, the maximum number of frames that have to be discarded equals $p_I + q - 1$.

Proof: As seen in Property 6, when a segment with an I frame is retrieved while the segment ahead of it (in the segment storage order) is not, the p_I frames belonging to the retrieved segment and ahead of the I frame (in the frame storage order) are discarded. The $(q-1)$ B that follow the I frame (in the frame storage order) are also discarded. However, all frames that follow (in the frame storage order) the last such B frame that has to be discarded can be successfully decoded. Hence, the result. □

Claim 1: When segments are retrieved consecutively (in the segment storage order) for the display of a piece of video, the first such retrieved segment should be one with an I frame in it.

Proof: An I frame is (either directly or indirectly) responsible for the successful decoding of the $(qm-1)$ frames that follow it in the frame storage order. A P frame is (either directly or indirectly) responsible for the successful decoding of the frames that follow it and that are before the next I frame (in the frame storage order). Hence, if a segment with no I frame in it is retrieved and the first segment before this (in the segment storage order) and containing an I/P frame is not retrieved, then all frames in this retrieved segment have to be discarded. □

Out of the d segment sets, we need to retrieve a certain number of segments so that the number of displayed frames equals a. Following Claim 1, the retrieval of consecutive segments for the display of a piece begins with a segment containing an I frame. The maximum number of pieces in d segment sets equals the number of segments containing an I frame, which from Property 4 is $d \times \frac{cs}{qm}$. From Properties 6 and 7, the number of retrieved frames that have to be discarded for each piece varies between $p_I + \min(s-p_I-1, q-1)$ and (p_I+q-1). A tradeoff exists between the number of pieces and the number of retrieved frames that have to be discarded. The more the number of pieces, the more is the number of retrieved frames that have to be discarded. In addition, (from Property 6) the earlier the position of the I frame in the first retrieved segment of a piece, the smaller is the number of frames that have to be discarded for that piece.

For the particular grouping illustrated in Figure 3, Figure 4 illustrates the case where $n = 30$ and $n_1 = 54$ fps. Out

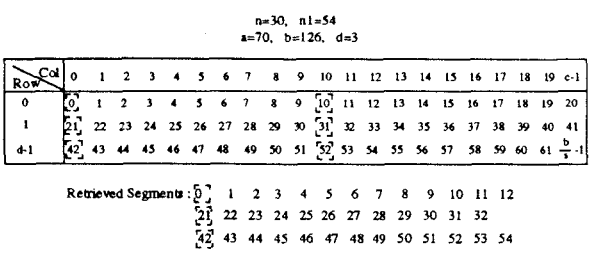

Figure 4: Retrieval during fast playback

of $b = 126$ frames, we need to display $a = 70$ frames. Three $(d = 3)$ segment sets need to be considered here and the maximum number of pieces in these $b/s = 63$ segments is six. The segments enclosed in dashed boxes are those that contain I frames. One possible set of retrieved segments with three displayed pieces is also shown in the figure.

4 Conclusions

In this paper, we proposed a generalized scheme that may be used to group frames into segments for any desired segment length. Due to the properties enjoyed by the minimal causal order, it was chosen to be the frame storage order. Closed form equations were provided that may be used to determine the frames that make up a segment. During fast playback, retrieval of some segments is skipped. The choice of segments to retrieve depends on the desired number of video pieces as well as on the maximum number of retrieved frames that can be discarded. Using the properties provided, a suitable retrieval scheme can be devised. Such grouping and retrieval schemes can be very useful in applications, such as VOD, that use stored video.

References

[1] Ming-Syan Chen, Dilip D. Kandlur, and Philip S. Yu. "Storage and Retrieval Methods to support fully interactive playout in a disk-array-based video server". *Multimedia Systems*, 3:126–135, 1995.

[2] Daniel Deloddere, Willem Verbiest, and Henri Verhille. "Interactive Video On Demand". *IEEE Communications Magazine*, pages 82–88, May 1994.

[3] Yurdaer N. Doganata and Asser N. Tantawi. "Making a cost-effective Video Server". *IEEE Multimedia*, 1994.

[4] Victor O. K. Li, Wanjiun Liao, Xiaoxin Qiu, and Eric W. M. Wong. "Performance model of interactive video-on-demand systems". *IEEE JSAC*, August 1996.

[5] Pramod Pancha and Magda El Zarki. "MPEG Coding for Variable Bit Rate Video Transmission". *IEEE Communications Magazine*, pages 54–66, May 1994.

[6] Senthil Sengodan and Victor O.K. Li. "A Quasi-static Retrieval Scheme for Interactive VOD Servers". *Computer Communications*, May 1997.

[7] Senthil Sengodan and Victor O.K. Li. "A Shared Buffer Architecture for Interactive VOD Servers". *IEEE INFOCOM*, April 1997.