File created: 14-Jan-2024 13:20:30 **{WMEDLEY}<lispusers>COMPARETEXT.;133**

edit by: rmk

changes to: (FNS IMCOMPARE.COLLECT.HASH.CHUNKS)

previous date: 14-Jan-2024 13:11:44 {WMEDLEY}<lispusers>COMPARETEXT.;132

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
(RPAQQ COMPARETEXTCOMS
       ((FNS COMPARETEXT COMPARETEXT.WINDOW COMPARETEXT.TSTREAM COMPARETEXT.SETSEL CHUNKNODELABEL
             IMCOMPARE.BOXNODE IMCOMPARE.CHUNKS IMCOMPARE.COLLECT.HASH.CHUNKS IMCOMPARE.DISPLAYGRAPH
             IMCOMPARE.HASH IMCOMPARE.MERGE.CONNECTED.CHUNKS IMCOMPARE.MERGE.UNCONNECTED.CHUNKS
             IMCOMPARE.SHOW.DIST IMCOMPARE.UPDATE.SYMBOL.TABLE)
        (FNS IMCOMPARE.LEFTBUTTONFN IMCOMPARE.MIDDLEBUTTONFN IMCOMPARE.COPYBUTTONFN)
        (FILES (SYSLOAD)
               GRAPHER REGIONMANAGER)
        (FNS TAIL1 TAIL2)
                                                                    ; Debugging
        (INITVARS (COMPARETEXT.ALLCHUNKS T)
               (COMPARETEXT.AUTOTEDIT T))
        (DECLARE%: EVAL@COMPILE DONTCOPY (RECORDS IMCOMPARE.CHUNK IMCOMPARE.SYMB)
               (FILES (LOADCOMP)
                      GRAPHER))))

(DEFINEQ
```

(COMPARETEXT

```
  [LAMBDA (FILE1 FILE2 HASH.TYPE CHUNKREGION FILELABELS TITLE TEXTWIDTH TEXTHEIGHT)
```

;; Edited 23-Jun-2022 17:13 by rmk: Clarified that the REGION is the region of the chunk window, added TEXTWIDTH and HEIGHT to specify the
;; size of each text window.

;; Edited 12-Jan-2022 16:32 by rmk

;; Edited 8-Nov-2021 08:44 by rmk

;; Edited 8-Jan-84 21:06 by mjs

;; Compares the two files, and produces a graph showing their corresponding chunks. The courseness of the 'chunking' is determined by
;; HASH.TYPE, which may be PARA, LINE, or WORD. HASH.TYPE = NIL defaults to PARA. The file difference graph is displayed at
;; CHUNKREGION. If CHUNKREGION = NIL, the user is asked to specify a region. If CHUNKREGION = T, a standard region is used.

```
    (SELECTQ HASH.TYPE
         ((PARA LINE WORD))
         (NIL (SETQ HASH.TYPE 'PARA))
         (ERROR (CONCAT "Unrecognize HASHTYPE " HASH.TYPE)))
    (LET [(FULLFILE1 (OR (GETSTREAM FILE1 'INPUT T)
                         (FINDFILE FILE1 T)))
          (FULLFILE2 (OR (GETSTREAM FILE2 'INPUT T)
                         (FINDFILE FILE2 T]
         (CL:UNLESS (AND FULLFILE1 FULLFILE2)
             (ERROR "Can't find both files" (LIST FILE1 FILE2)))
         (IMCOMPARE.CHUNKS (create IMCOMPARE.CHUNK
                                   FILENAME _ FULLFILE1
                                   FILEPTR _ 0)
                (create IMCOMPARE.CHUNK
                       FILENAME _ FULLFILE2
                       FILEPTR _ 0)
                HASH.TYPE CHUNKREGION FILELABELS TITLE TEXTWIDTH TEXTHEIGHT])
```

(COMPARETEXT.WINDOW

```
  [LAMBDA (GRAPH CHUNKREGION TITLE)                                 ; Edited 23-Jun-2022 16:56 by rmk
                                                                    ; Edited 25-Feb-2022 14:34 by rmk
                                                                    ; Edited 19-Feb-2022 12:01 by rmk
                                                                    ; Edited  2-Feb-2022 17:29 by rmk
                                                                    ; Edited 23-Jan-2022 18:18 by rmk
                                                                    ; Edited 12-Jan-2022 10:06 by rmk
                                                                    ; Edited 22-Dec-2021 15:51 by rmk
```

;; Set up the graph WINDOW. If REGION isn't provided we prompt with a region that is wide enough for the graph and high enough for at least an
;; initial segment.

```
    (LET [WINDOW GRAPHREGION WIDTH HEIGHT (FILEPREFIX (CAR (GRAPHERPROP GRAPH 'FILELABELS]
         (SETQ GRAPHREGION (GRAPHREGION GRAPH))
         [SETQ WIDTH (IMIN SCREENWIDTH (IPLUS (TIMES 2 WBorder)
                                              (FETCH (REGION WIDTH) OF GRAPHREGION]
         [SETQ HEIGHT (IMIN 200 (IPLUS (FETCH (REGION HEIGHT) OF GRAPHREGION)
                                       (ITIMES 2 (FONTHEIGHT DEFAULTFONT]
         (SETQ CHUNKREGION (if (EQ CHUNKREGION T)
                               then (create REGION
                                            LEFT _ 25
                                            BOTTOM _ 25
                                            WIDTH _ 500
                                            HEIGHT _ 150)
                              elseif (REGIONP CHUNKREGION)
```

```
                        elseif (POSITIONP CHUNKREGION)
                           THEN ;; This is a reference position providing the horizontal midpoint of the graph region and the top
                                (RELCREATEREGION WIDTH HEIGHT 'LEFT 'TOP (IDIFFERENCE (FETCH (POSITION
                                                                                             XCOORD)
                                                                                      OF CHUNKREGION)
                                                                              (IQUOTIENT WIDTH 2))
                                                 (FETCH (POSITION YCOORD) OF CHUNKREGION))
                        ELSE (RELCREATEREGION WIDTH HEIGHT 'RIGHT 'TOP CHUNKREGION)))
     [SETQ WINDOW (CREATEW CHUNKREGION (OR TITLE (CONCAT "Compare text" (CL:IF FILEPREFIX
                                                                              (CONCAT " of " FILEPREFIX)
                                                                              "")
                                                                " showing "
                                                                (CL:IF (GRAPHERPROP GRAPH 'ALLCHUNKS)
                                                                    "all"
                                                                    "only different")
                                                                " chunks, hashed by "
                                                                (SELECTQ (GRAPHERPROP GRAPH 'HASH.TYPE)
                                                                     (PARA "paragraph")
                                                                     (LINE "line")
                                                                     (WORD "word")
                                                                     (SHOULDNT]
     (GETPROMPTWINDOW WINDOW)
     (CL:WHEN (EQ WIDTH (FETCH (REGION WIDTH) OF (WINDOWREGION WINDOW)))
         (WINDOWPROP WINDOW 'MAXSIZE (CONS WIDTH MAX.SMALLP)))
     (GETPROMPTWINDOW WINDOW)
     WINDOW])
```

(COMPARETEXT.TSTREAM

```
  [LAMBDA (NODE WINDOW INCOL1)                                    ; Edited  2-Nov-2022 00:11 by rmk
                                                                  ; Edited 23-Jun-2022 17:20 by rmk
                                                                  ; Edited 18-Feb-2022 17:05 by rmk
                                                                  ; Edited 30-Jan-2022 09:03 by rmk
                                                                  ; Edited 28-Jan-2022 22:37 by rmk
    ;; Returns the text stream for the chunk column in the graphwindow WINDOW, on the left if INCOL1.  If the windows are automatic, they are lined
    ;; up under the middle of WINDOW.

    (DECLARE (USEDFREE COMPARETEXT.AUTOTEDIT))
    (LET [TWINDOW REGION REGIONARGS TEXTWIDTH TEXTHEIGHT (GRAPH (WINDOWPROP WINDOW 'GRAPH))
               (NODEID (FETCH (GRAPHNODE NODEID) OF NODE))
               (TSTREAM (WINDOWPROP WINDOW (CL:IF INCOL1
                                               'COL1TSTREAM
                                               'COL2TSTREAM)]
         (CL:UNLESS (AND TSTREAM (OPENWP (WFROMDS TSTREAM)))
             ;; First time, we have the graph but we don't yet have the TEDIT stream and window
             (SETQ TEXTWIDTH (OR (GRAPHERPROP GRAPH 'TEXTWIDTH)
                                 700))
             (SETQ TEXTHEIGHT (OR (GRAPHERPROP GRAPH 'TEXTHEIGHT)
                                  600))
             (SETQ REGIONARGS (LIST TEXTWIDTH TEXTHEIGHT (CL:IF INCOL1
                                                             'RIGHT
                                                             'LEFT)
                                    'TOP
                                    `(,WINDOW 0.5 ,(CL:IF INCOL1
                                                       -1
                                                       1))
                                    `(,WINDOW BOTTOM -2)
                                    NIL))
             (SETQ REGION (CL:IF COMPARETEXT.AUTOTEDIT
                                 (RELCREATEREGION REGIONARGS)
                                 (RELGETREGION REGIONARGS)))
             ;; If the CAR is a FIXP, this is a chunk node.  Otherwise, it is one of the two file-name column headers.
             [SETQ TSTREAM (TEXTSTREAM (TEDIT (CL:IF (FIXP (CAR NODEID))
                                                     (FETCH (IMCOMPARE.CHUNK FILENAME) of NODEID)
                                                     NODEID)
                                              REGION NIL `(READONLY T LEAVETTY T]
             (SETQ TWINDOW (WFROMDS TSTREAM))
             (WINDOWPROP WINDOW (CL:IF INCOL1
                                    'COL1TSTREAM
                                    'COL2TSTREAM)
                    TSTREAM)
             [WINDOWPROP TWINDOW 'TITLE (CL:IF INCOL1
                                            (CADR (GRAPHERPROP GRAPH 'FILELABELS))
                                            (CADDR (GRAPHERPROP GRAPH 'FILELABELS)))]
             (MOVEWITH TWINDOW WINDOW)
             (CLOSEWITH TWINDOW WINDOW))
         TSTREAM])
```

(COMPARETEXT.SETSEL

```
  [LAMBDA (TSTREAM NODE)                                          ; Edited  2-Nov-2022 10:07 by rmk
                                                                  ; Edited 25-Dec-2021 10:52 by rmk
    (LET* ((CHUNK (FETCH (GRAPHNODE NODEID) OF NODE))
           (FILEPTR (fetch (IMCOMPARE.CHUNK FILEPTR) of CHUNK)))
```

```
        ;; The first selection just makes sure that at least 25 characters before the chunk will be on the screen.  The second causes only the
        ;; characters of the actual chunk to be underlined and shown.
        (TEDIT.SETSEL TSTREAM (IMAX 1 (IDIFFERENCE FILEPTR 25))
               0
               'LEFT)
        (TEDIT.NORMALIZECARET TSTREAM)
        (TEDIT.SETSEL TSTREAM FILEPTR (fetch (IMCOMPARE.CHUNK CHUNKLENGTH) of CHUNK)
               'LEFT)
        (TEDIT.NORMALIZECARET TSTREAM])
```

(CHUNKNODELABEL

```
  [LAMBDA (CHUNK MIN.LENGTH EXTENDER)                                    ; Edited 25-Dec-2021 11:56 by rmk
                                                                         ; Edited 13-Dec-2021 21:18 by rmk
                                                                         (* mjs "30-Dec-83 15:11")
```

```
    ;; Label for CHUNK  is at least MIN.LENGTH characters long, by concatenating the first character of EXTENDER (or space, if not given) to the front
    (LET ((FILEPTR (fetch (IMCOMPARE.CHUNK FILEPTR) of CHUNK))
          (LENGTH (fetch (IMCOMPARE.CHUNK CHUNKLENGTH) of CHUNK))
          X)
         (SETQ X (CONCAT FILEPTR ":" LENGTH))
         (AND NIL (IF (ILESSP (NCHARS X)
                              MIN.LENGTH)
                      THEN (CONCAT (ALLOCSTRING (IDIFFERENCE MIN.LENGTH (NCHARS X))
                                                (CL:IF EXTENDER
                                                    (NTHCHAR EXTENDER 1)
                                                    " "))
                                   X)
                      ELSE X))
         X])
```

(IMCOMPARE.BOXNODE

```
  [LAMBDA (WINDOW NODE1 NODE2)
```

```
    ;; Edited 20-May-2022 16:35 by rmk: Invert nodes rather than FLIPNODES, so they stay inverted when scrolled
    ;; Edited 25-Dec-2021 12:01 by rmk                                    (* rmk%: "14-Dec-84 13:40")
    ;; Marks NODE1 and NODE2 as having been selected, removing marks on previous nodes.
    (LET [(LASTNODES (WINDOWPROP WINDOW 'LASTNODES]                      ; FLIPNODE ?
         (CL:WHEN (CAR LASTNODES)
             (RESET/NODE/LABELSHADE (CAR LASTNODES)
                    'INVERT WINDOW))
         (CL:WHEN (CADR LASTNODES)
             (RESET/NODE/LABELSHADE (CADR LASTNODES)
                    'INVERT WINDOW))
         (CL:WHEN NODE1
             (RESET/NODE/LABELSHADE NODE1 'INVERT WINDOW))
         (CL:WHEN NODE2
             (RESET/NODE/LABELSHADE NODE2 'INVERT WINDOW))
         (WINDOWPROP WINDOW 'LASTNODES (LIST NODE1 NODE2])
```

(IMCOMPARE.CHUNKS

```
  [LAMBDA (CHUNK1 CHUNK2 HASH.TYPE CHUNKREGION FILELABELS TITLE TEXTWIDTH TEXTHEIGHT)
                                                                         ; Edited 23-Jun-2022 17:13 by rmk
                                                                         ; Edited 12-Jan-2022 10:06 by rmk
                                                                         ; Edited 23-Dec-2021 00:02 by rmk
                                                                         ; Edited  8-Sep-1984 00:06 by rmk
```

```
    ;; This is the main text-comparison function.  It compares the text in the two chunks <which may be small pieces of files, or entire files> and
    ;; produces a graph showing how the sub-chunks of the two main chunks are related.  The two main chunks may be in the same file, and the file
    ;; may actually be an open Tedit textstream.  The main chunks are broken down according to HASH.TYPE, which may be PARA <chunk by
    ;; paragraph>, LINE, WORD,  or  PARA.  The file difference graph is displayed at REGION.
    ;; This text comparison algorithm is originally from the article 'A Technique for Isolating Differences Between Files' by Paul Heckel, in CACM, V21,
    ;; #4, April 1978 --- major difference is that I use lists instead of arrays
    ;;
    ;; Collect lists of chunks from each of the main chunks, dividing them according to HASH.TYPE.  We start with whole-file chunk. but this works also
    ;; for a chunk that corresponds to a subsection of a file.
    (LET ((CHUNK.SYMBOL.TABLE (HASHARRAY 500))
          (CHUNKLIST1 (IMCOMPARE.COLLECT.HASH.CHUNKS CHUNK1 HASH.TYPE))
          (CHUNKLIST2 (IMCOMPARE.COLLECT.HASH.CHUNKS CHUNK2 HASH.TYPE)))
        ;; Update the chunk symbol table.  For each hash value, this table records the number of file1 chunks with that hash value, the number of
        ;; file2 chunks with that value, and a pointer to a tail of CHUNKLIST2 (not to a chunk itself).
        (IMCOMPARE.UPDATE.SYMBOL.TABLE CHUNKLIST1 CHUNK.SYMBOL.TABLE NIL)
        (IMCOMPARE.UPDATE.SYMBOL.TABLE CHUNKLIST2 CHUNK.SYMBOL.TABLE T)
        ;; For every file1 chunk whose hash value matches EXACTLY ONE file2 chunk's value, 'connect' it to the file2 chunk by setting the file1
        ;; chunk's OTHERCHUNK field to point to the appropriate tail of the file1 chunk list <not the chunk directly>.  Also, make sure that
        ;; OTHERCHUNK of the matching file1 chunk is non-NIL, so that unconnected file1 chunks will be merged correctly.
        (for C1 in CHUNKLIST1 bind SYMB do (SETQ SYMB (GETHASH (fetch (IMCOMPARE.CHUNK HASHVALUE) of C1)
                                                               CHUNK.SYMBOL.TABLE))
                                           (if (AND (EQ 1 (fetch (IMCOMPARE.SYMB NEWCOUNT) of SYMB))
```

```
                                                    (EQ 1 (fetch (IMCOMPARE.SYMB OLDCOUNT) of SYMB)))
                                                then (replace (IMCOMPARE.CHUNK OTHERCHUNK) of C1
                                                        with (fetch (IMCOMPARE.SYMB OLDPTR) of SYMB))
                                                     (replace (IMCOMPARE.CHUNK OTHERCHUNK)
                                                        of (CAR (fetch (IMCOMPARE.SYMB OLDPTR) of SYMB))
                                                        with T)))
```

;; Merge connected chunks forward

```
(IMCOMPARE.MERGE.CONNECTED.CHUNKS CHUNKLIST1 NIL)
```

;; Merge connected chunks backwards

```
(SETQ CHUNKLIST1 (DREVERSE CHUNKLIST1))
(SETQ CHUNKLIST2 (DREVERSE CHUNKLIST2))
(IMCOMPARE.MERGE.CONNECTED.CHUNKS CHUNKLIST1 T)
(SETQ CHUNKLIST1 (DREVERSE CHUNKLIST1))
(SETQ CHUNKLIST2 (DREVERSE CHUNKLIST2))
```

;; Merge unconnected chunks

```
(IMCOMPARE.MERGE.UNCONNECTED.CHUNKS CHUNKLIST1)
(IMCOMPARE.MERGE.UNCONNECTED.CHUNKS CHUNKLIST2)
```

;; The file comparison is complete. Format and display the file difference graph

```
(IMCOMPARE.DISPLAYGRAPH CHUNK1 CHUNK2 HASH.TYPE CHUNKREGION CHUNKLIST1 CHUNKLIST2 FILELABELS TITLE
       TEXTWIDTH TEXTHEIGHT])
```

(IMCOMPARE.COLLECT.HASH.CHUNKS

```
[LAMBDA (CHUNK HASH.TYPE)                                   ; Edited 14-Jan-2024 13:20 by rmk
                                                            ; Edited 18-Oct-2023 17:45 by rmk
                                                            ; Edited 20-Jan-2022 23:09 by rmk
                                                            ; Edited 24-Dec-2021 22:30 by rmk
                                                            ; Edited 13-Dec-2021 16:32 by rmk
                                                            ; Edited 23-Dec-98 16:54 by rmk:
                                                            (* mjs " 8-Jan-84 20:57")
```

;;; Returns a list of the chunks inside CHUNK as hashed of type HASH.TYPE. Presumably CHUNK is is higher on the ranking PARA > LINE >. WORD.
;;; The initial CHUNK covers the whole file, middle-mouse refinement-chunks cover only subsections.

;; It is overkill to open raw text streams as TEDIT stream. So we open, test for TEDIT and if so, close and reoopen.

```
(RESETLST
    (BIND (FILENAME _ (fetch (IMCOMPARE.CHUNK FILENAME) of CHUNK))
          STREAM ENDPOS FIRST [RESETSAVE [SETQ STREAM (OPENSTREAM FILENAME 'INPUT 'OLD '((ENDOFSTREAMOP
                                                                                          NILL]
                                                 '(PROGN (CLOSEF? OLDVALUE]
                                       (CL:WHEN (TEDIT.FORMATTEDFILEP STREAM)
                                                                    ; The OBJECTCHAR is produced in place of image objects
                                           [RESETSAVE [SETQ STREAM (OPENTEXTSTREAM STREAM NIL NIL NIL
                                                                                   `(OBJECTBYTE ,(CHARCODE *]
                                                  '(PROGN (CLOSEF? OLDVALUE])
                                       (SETFILEINFO STREAM 'EOL 'ANY)
                                       (CL:UNLESS (fetch (IMCOMPARE.CHUNK CHUNKLENGTH) of CHUNK)
                                           ;; For TEDIT files, the character length isn't known until after text-opening
                                           (REPLACE (IMCOMPARE.CHUNK CHUNKLENGTH) of CHUNK WITH (GETFILEINFO
                                                                                                STREAM
                                                                                                'LENGTH)))
                                       (SETFILEPTR STREAM (fetch (IMCOMPARE.CHUNK FILEPTR) of CHUNK))
                                       (SETQ ENDPOS (IPLUS (fetch (IMCOMPARE.CHUNK FILEPTR) of CHUNK)
                                                           (fetch (IMCOMPARE.CHUNK CHUNKLENGTH) of CHUNK)))
          WHILE (SETQ CHUNK (IMCOMPARE.HASH STREAM HASH.TYPE ENDPOS)) COLLECT (REPLACE (IMCOMPARE.CHUNK
                                                                                        FILENAME)
                                                                                  OF CHUNK WITH FILENAME)
                                                                               CHUNK))])
```

(IMCOMPARE.DISPLAYGRAPH

```
[LAMBDA (CHUNK1 CHUNK2 HASH.TYPE CHUNKREGION CHUNKLIST1 CHUNKLIST2 FILELABELS TITLE TEXTWIDTH TEXTHEIGHT)
                                                            ; Edited 23-Jun-2022 17:13 by rmk
                                                            ; Edited 12-Jan-2022 09:58 by rmk
                                                            ; Edited 27-Dec-2021 11:58 by rmk
                                                            ; Edited 23-Dec-2021 00:14 by rmk
                                                            (* mjs "11-Jul-85 09:10")
```

;; Format and display the graph

```
(DECLARE (USEDFREE COMPARETEXT.ALLCHUNKS))
(LET ((FULLFILE1 (fetch (IMCOMPARE.CHUNK FILENAME) of CHUNK1))
      (FULLFILE2 (fetch (IMCOMPARE.CHUNK FILENAME) of CHUNK2))
      FILE1LABEL FILE2LABEL FILEPREFIX 2TO1MAP (BORDERSIZE 1)
      NODES1 NODES2 COL1HEADER COL1X COL2HEADER COL2X YINCREMENT GRAPH TEMP1)
```

;; Create the nodes for the column headers

```
(SETQ FILE1LABEL (OR (CAR (LISTP FILELABELS))
                     FULLFILE1))
(SETQ FILE2LABEL (OR (CADR (LISTP FILELABELS))
                     FULLFILE2))
(CL:WHEN (SETQ FILEPREFIX (FB.GREATEST.PREFIX FILE1LABEL FILE2LABEL))
    [SETQ FILE1LABEL (SUBSTRING FILE1LABEL (ADD1 (NCHARS FILEPREFIX]
```

```
    [SETQ FILE2LABEL (SUBSTRING FILE2LABEL (ADD1 (NCHARS FILEPREFIX])
(SETQ COL1X (IQUOTIENT (STRINGWIDTH FILE1LABEL DEFAULTFONT)
                       2))
(SETQ COL1HEADER (NODECREATE FULLFILE1 FILE1LABEL (CREATEPOSITION COL1X 0)
                             NIL NIL DEFAULTFONT -2))
[SETQ COL2X (IPLUS COL1X (IMAX 100 (IPLUS COL1X 30 (IQUOTIENT (STRINGWIDTH FILE2LABEL DEFAULTFONT)
                                                              2]
(SETQ COL2HEADER (NODECREATE FULLFILE2 FILE2LABEL (CREATEPOSITION COL2X 0)
                             NIL NIL DEFAULTFONT -2))
```

;; It would be nice to get corresponding chunks at the same positions in their lists, so that equality lines will be horizontal. Different numbers ;; of inserts above can throw that off, we try to insert NIL spaces to even things up.

```
[FOR C1TAIL C1 O1 ON CHUNKLIST1 AS C2TAIL C2 ON CHUNKLIST2
   EACHTIME (SETQ C1 (CAR C1TAIL))
         (SETQ C2 (CAR C2TAIL))
         (SETQ O1 (CAR (FETCH OTHERCHUNK OF C1)))
   UNLESS (EQ C2 O1) DO (IF (AND O1 (EQ O1 (CADR C2TAIL)))
                            THEN ;; We push NIL into the C1TAIL cell that C1 formerly occupied, move C1 down
                                 (ATTACH NIL C1TAIL)
                              ELSEIF [EQ C2 (CAR (FETCH OTHERCHUNK OF (SETQ C1 (CADR C1TAIL]
                                THEN (ATTACH NIL C2TAIL)  ; OTHERCHUNK is the tail that contains C2, so it also has to be
                                                          ; updated.
                                     (REPLACE OTHERCHUNK OF C1 WITH (CDR C2TAIL)))
                          ;; Make them run out at the same time.
                          (IF (AND (CDR C1TAIL)
                                   (NULL (CDR C2TAIL)))
                              THEN (RPLACD C2TAIL (CONS))
                            ELSEIF (AND (CDR C2TAIL)
                                        (NULL (CDR C1TAIL)))
                              THEN (RPLACD C1TAIL (CONS]
[SETQ YINCREMENT (IMINUS (IPLUS 2 (ITIMES 2 BORDERSIZE)
                                  (FONTPROP DEFAULTFONT 'HEIGHT]
```

;; Collect new-chunk graph nodes, while accumulating 2TO1MAP, assoc list from file2 chunks to file1 chunks. We skip the NILs inserted ;; above (although Y increments).

```
[SETQ NODES1 (for C1 C2 in CHUNKLIST1 as Y from YINCREMENT by YINCREMENT
                collect (CL:WHEN C1
                           (CL:WHEN (SETQ C2 (CAR (fetch (IMCOMPARE.CHUNK OTHERCHUNK) of C1)))
                               (PUSH 2TO1MAP (CONS C2 C1)))
                                                          ; Start out with 2 point white border, so we can invert it
                           (NODECREATE C1 (CHUNKNODELABEL C1 10)
                                  (CREATEPOSITION COL1X Y)
                                  (CL:WHEN C2 (CONS C2))
                                  NIL DEFAULTFONT -2))]
[SETQ NODES2 (for C2 C1 in CHUNKLIST2 as Y from YINCREMENT by YINCREMENT
                collect (CL:WHEN C2
                           (SETQ C1 (CDR (ASSOC C2 2TO1MAP)))
                           (NODECREATE C2 (CHUNKNODELABEL C2 10 (AND NIL "-"))
                                  (CREATEPOSITION COL2X Y)
                                  NIL
                                  (CL:WHEN C1 (CONS C1))
                                  DEFAULTFONT -2))]
```

;; Now eliminate all the C1/C2 node pairs that are at the same Yposition. Those would just have uninformative horizontal lines representing ;; no differences. Maybe this can be done on the fly--don't construct such pairs--but that will come later. The node

```
(IF COMPARETEXT.ALLCHUNKS
    THEN (SETQ NODES1 (DREMOVE NIL NODES1))
         (SETQ NODES2 (DREMOVE NIL NODES2))
  ELSE ;; The nodes in both lists correspond, with NILs padding where needed. We can simplify the picture if we take out equivalent
       ;; chunks, otherwise we show all their horizontal lines.
       (FOR N1 KEPT1 KEPT2 (YPOS _ YINCREMENT) IN NODES1 AS N2 IN NODES2
          UNLESS [AND N1 N2 (EQ (FETCH NODEID OF N2)
                                (CAR (FETCH OTHERCHUNK OF (FETCH NODEID OF N1]
          DO (CL:WHEN N1
                 (PUSH KEPT1 N1)
                 (REPLACE YCOORD OF (FETCH NODEPOSITION OF N1) WITH YPOS))
             (CL:WHEN N2
                 (PUSH KEPT2 N2)
                 (REPLACE YCOORD OF (FETCH NODEPOSITION OF N2) WITH YPOS))
             (ADD YPOS YINCREMENT)
          FINALLY (SETQ NODES1 KEPT1)
                  (SETQ NODES2 KEPT2)))
```

;; Keep column xcords so leftbutton can tell a node's column, keep labels for new middle mouse graph

```
[SETQ GRAPH (create GRAPH
                    DIRECTEDFLG _ T
                    SIDESFLG _ T
                    GRAPHNODES _ (NCONC (LIST COL1HEADER)
                                        NODES1
                                        (LIST COL2HEADER)
                                        NODES2)
                    GRAPH.PROPS _ `(HASH.TYPE ,HASH.TYPE FILELABELS (,FILEPREFIX ,FILE1LABEL
                                                                                 ,FILE2LABEL)
                                                                    COL1X
```

```
                                                ,COL1X COL2X ,COL2X ALLCHUNKS ,COMPARETEXT.ALLCHUNKS TEXTWIDTH
                                                ,TEXTWIDTH TEXTHEIGHT ,TEXTHEIGHT]
        (SHOWGRAPH GRAPH (COMPARETEXT.WINDOW GRAPH CHUNKREGION TITLE)
               (FUNCTION IMCOMPARE.LEFTBUTTONFN)
               (FUNCTION IMCOMPARE.MIDDLEBUTTONFN)
               T NIL])
```

(IMCOMPARE.HASH

```
  [LAMBDA (STREAM HASH.TYPE ENDPOS)                                    ; Edited 18-Oct-2023 17:44 by rmk
                                                                       ; Edited 19-Dec-2021 09:07 by rmk
                                                                       ; Edited 15-Dec-2021 15:58 by rmk
                                                                       ; Edited 13-Dec-2021 16:35 by rmk
                                                                       ; Edited 23-Dec-98 16:58 by rmk:
    ;; IMCOMPARE.HASH automatically stops before reading char number EOF.PTR.

    ;; Returns an IMCOMPARE.CHUNK containing the hash value, the file pointer of the beginning of the chunk, the length of the chunk, and the
    ;; fullname of the stream

    ;; Note: Most of the time in COMPARETEXT is spent reading in and hashing chunks, so this function was optimizes for speed, at the expense of
    ;; length

    (LET ((STARTPOS (GETFILEPTR STREAM))
          (HASHNUM 0)
          C NBYTES)
         (DECLARE (SPECVARS NBYTES))
         (SETQ NBYTES (IDIFFERENCE ENDPOS STARTPOS))                   ; \INCCODE counts down. We reach NBYTES only on the chunk

         ;; Don't hash on white space

         (CL:WHEN (IGREATERP NBYTES 0)
             (SELECTQ HASH.TYPE
                  (PARA                                                ; Paragraph chunks end with two consecutive EOL's.
                        (BIND EOLSEEN WHILE (IGREATERP NBYTES 0)
                           DO (SELCHARQ (SETQ C (\INCCODE.EOLC STREAM NIL 'NBYTES NBYTES))
                                    ((EOL NIL)
                                         (CL:WHEN EOLSEEN (RETURN))
                                         (SETQ EOLSEEN T)            ; Skip the NIL SETQ below
                                         (GO $$ITERATE))
                                    ((SPACE TAB))
                                    (SETQ HASHNUM (ROT (ROT (ROT (LOGXOR HASHNUM C)
                                                                 1 16)
                                                            1 16)
                                                       1 16)))
                              (SETQ EOLSEEN NIL)))
                  (LINE                                                ; Line chunks end on EOL.
                        [WHILE (IGREATERP NBYTES 0) DO (SELCHARQ (SETQ C (\INCCODE.EOLC STREAM NIL 'NBYTES
                                                                           NBYTES))
                                                         (EOL (RETURN))
                                                         ((SPACE TAB))
                                                         (SETQ HASHNUM (ROT (ROT (ROT (LOGXOR HASHNUM C)
                                                                                      1 16)
                                                                                 1 16)
                                                                            1 16])
                  (WORD                                                ; word chunks end on any white space
                        [WHILE (IGREATERP NBYTES 0) DO (SELECTQ (SETQ C (\INCCODE.EOLC STREAM NIL 'NBYTES NBYTES
                                                                                       ))
                                                         ((SPACE EOL TAB)
                                                             (RETURN))
                                                         (SETQ HASHNUM (ROT (ROT (ROT (LOGXOR HASHNUM C)
                                                                                      1 16)
                                                                                 1 16)
                                                                            1 16])
                  (SHOULDNT))                                          ; flush all white space before next chunk
             (WHILE (IGREATERP NBYTES 0) DO (SELCHARQ (\INCCODE.EOLC STREAM NIL 'NBYTES NBYTES)
                                                      ((EOL SPACE TAB))
                                                      (RETURN)))
             (CREATE IMCOMPARE.CHUNK
                    HASHVALUE _ HASHNUM
                    FILEPTR _ STARTPOS
                    CHUNKLENGTH _ (IDIFFERENCE (GETFILEPTR STREAM)
                                               STARTPOS)))])
```

(IMCOMPARE.MERGE.CONNECTED.CHUNKS

```
  [LAMBDA (NEW.CHUNK.LIST BACKWARDS.FLG)                               (* mjs " 6-Jan-84 10:35")
    (while NEW.CHUNK.LIST bind NEW.CHUNK OLD.CHUNK.PTR
       do (SETQ NEW.CHUNK (CAR NEW.CHUNK.LIST))
          (SETQ OLD.CHUNK.PTR (fetch (IMCOMPARE.CHUNK OTHERCHUNK) of NEW.CHUNK))
          (if [OR (NULL (CDR NEW.CHUNK.LIST))
                  (NULL OLD.CHUNK.PTR)
                  (NULL (CDR OLD.CHUNK.PTR))
                  (NOT (EQP (fetch (IMCOMPARE.CHUNK HASHVALUE) of (CADR NEW.CHUNK.LIST))
                            (fetch (IMCOMPARE.CHUNK HASHVALUE) of (CADR OLD.CHUNK.PTR]
            then (SETQ NEW.CHUNK.LIST (CDR NEW.CHUNK.LIST))
          else

          (* next chunks have same hash, so "murge" them into current chunks by adding their chunk lengths to the current chunks,
          and splicing out the next chunks)
```

```
                [replace (IMCOMPARE.CHUNK CHUNKLENGTH) of NEW.CHUNK with (IPLUS (fetch (IMCOMPARE.CHUNK CHUNKLENGTH
                                                                                       )
                                                                                    of NEW.CHUNK)
                                                                                 (fetch (IMCOMPARE.CHUNK CHUNKLENGTH
                                                                                       )
                                                                                    of (CADR NEW.CHUNK.LIST]
                [replace (IMCOMPARE.CHUNK CHUNKLENGTH) of (CAR OLD.CHUNK.PTR) with (IPLUS (fetch (IMCOMPARE.CHUNK
                                                                                                  CHUNKLENGTH)
                                                                                               of (CAR OLD.CHUNK.PTR
                                                                                                     ))
                                                                                            (fetch (IMCOMPARE.CHUNK
                                                                                                  CHUNKLENGTH)
                                                                                               of (CADR
                                                                                                     OLD.CHUNK.PTR
                                                                                                     ]
                [if BACKWARDS.FLG
                    then                                                    (* if the list is backwards, copy next fileptr)
                         (replace (IMCOMPARE.CHUNK FILEPTR) of NEW.CHUNK with (fetch (IMCOMPARE.CHUNK FILEPTR)
                                                                                   of (CADR NEW.CHUNK.LIST)))
                         (replace (IMCOMPARE.CHUNK FILEPTR) of (CAR OLD.CHUNK.PTR) with (fetch (IMCOMPARE.CHUNK
                                                                                                 FILEPTR)
                                                                                              of (CADR OLD.CHUNK.PTR]
                                                                            (* splice chunks out of new and old list)
                (RPLACD NEW.CHUNK.LIST (CDDR NEW.CHUNK.LIST))
                (RPLACD OLD.CHUNK.PTR (CDDR OLD.CHUNK.PTR])
```

(IMCOMPARE.MERGE.UNCONNECTED.CHUNKS

```
  [LAMBDA (CHUNK.LST)                                                   (* mjs " 5-JAN-84 13:58")
    (while CHUNK.LST bind CHUNK do (SETQ CHUNK (CAR CHUNK.LST))
                                   (if (OR (NULL (CDR CHUNK.LST))
                                           (fetch (IMCOMPARE.CHUNK OTHERCHUNK) of CHUNK)
                                           (fetch (IMCOMPARE.CHUNK OTHERCHUNK) of (CADR CHUNK.LST)))
                                       then (SETQ CHUNK.LST (CDR CHUNK.LST))
                                     else                               (* both current chunk and next chunk have no OTHERCHUNK,
                                                                        so merge them)
                                          [replace (IMCOMPARE.CHUNK CHUNKLENGTH) of CHUNK
                                             with (IPLUS (fetch (IMCOMPARE.CHUNK CHUNKLENGTH) of CHUNK)
                                                         (fetch (IMCOMPARE.CHUNK CHUNKLENGTH) of (CADR CHUNK.LST]
                                                                        (* splice chunks out of new and old list)
                                          (RPLACD CHUNK.LST (CDDR CHUNK.LST])
```

(IMCOMPARE.SHOW.DIST

```
  [LAMBDA (LST MAX)                                                     (* mjs "30-Dec-83 15:13")
    (PROG ((WINDOW (CREATEW))
           MAX.Y X MAX.X)
          (SETQ MAX.X (WINDOWPROP WINDOW 'WIDTH))
          (SETQ MAX.Y (WINDOWPROP WINDOW 'HEIGHT))
          (for SAMPLE in LST do (SETQ X (FTIMES MAX.X (FQUOTIENT SAMPLE MAX)))
                                (DRAWLINE X 0 X MAX.Y 1 'PAINT WINDOW])
```

(IMCOMPARE.UPDATE.SYMBOL.TABLE

```
  [LAMBDA (CHUNK.LIST CHUNK.SYMBOL.TABLE OLD.CHUNK.FLG)                 (* mjs " 8-Jan-84 21:01")

          (* * update the chunk symbol table. For each hash value, this table records the number of "new" chunks with that hash
          value, the number of "old" chunks with that value, and a pointer to the place in OLD.CHUNK.LIST <not to an OLD chunk
          itself>.)

    (for CHUNK.PTR on CHUNK.LIST bind CHUNK SYMB
       do (SETQ CHUNK (CAR CHUNK.PTR))
          (SETQ SYMB (if (GETHASH (fetch (IMCOMPARE.CHUNK HASHVALUE) of CHUNK)
                                  CHUNK.SYMBOL.TABLE)
                         else (PUTHASH (fetch (IMCOMPARE.CHUNK HASHVALUE) of CHUNK)
                                       (create IMCOMPARE.SYMB
                                               NEWCOUNT _ 0
                                               OLDCOUNT _ 0
                                               OLDPTR _ NIL)
                                       CHUNK.SYMBOL.TABLE)))
          (if OLD.CHUNK.FLG
              then                                                      (* increment old-chunk count)
                   (replace (IMCOMPARE.SYMB OLDCOUNT) of SYMB with (ADD1 (fetch (IMCOMPARE.SYMB OLDCOUNT)
                                                                                of SYMB)))

          (* smash old-chunk pointer. Note that it must point to the LIST of old-chunks, rather than to the individual one)

                   (replace (IMCOMPARE.SYMB OLDPTR) of SYMB with CHUNK.PTR)
            else                                                        (* increment new-chunk count)
                 (replace (IMCOMPARE.SYMB NEWCOUNT) of SYMB with (ADD1 (fetch (IMCOMPARE.SYMB NEWCOUNT) of SYMB])

)

(DEFINEQ

(IMCOMPARE.LEFTBUTTONFN
```

```
[LAMBDA (NODE WINDOW)
  ;; Edited  1-Nov-2022 22:29 by rmk
  ;; Edited 23-Jun-2022 22:50 by rmk: Turn off previous selection before turning on new one
  ;; Edited 25-Dec-2021 23:29 by rmk
  ;; Edited 22-Dec-2021 21:41 by rmk
  ;; Edited 18-Dec-2021 13:02 by rmk                                    (* mjs " 2-Apr-85 14:21")
  (CL:WHEN NODE
      (LET ([INCOL1 (EQ (GRAPHERPROP (WINDOWPROP WINDOW 'GRAPH)
                                     'COL1X)
                        (FETCH (POSITION XCOORD) OF (FETCH (GRAPHNODE NODEPOSITION) OF NODE]
            TSTREAM)
           ;; Turn off any previous selection
           (CL:WHEN (SETQ TSTREAM (WINDOWPROP WINDOW 'COL1TSTREAM))
                  (TEDIT.SHOWSEL TSTREAM))
           (CL:WHEN (SETQ TSTREAM (WINDOWPROP WINDOW 'COL2TSTREAM))
                  (TEDIT.SHOWSEL TSTREAM))
           (IF (FIXP (CAR (fetch (GRAPHNODE NODEID) of NODE)))
               THEN (IMCOMPARE.BOXNODE WINDOW NODE (FOR N (YPOS _ (FETCH YCOORD OF (FETCH NODEPOSITION
                                                                                          OF NODE)))
                                                                  IN (FETCH GRAPHNODES OF (WINDOWPROP WINDOW 'GRAPH))
                                                                  UNLESS (EQ N NODE)
                                                                  WHEN (EQ YPOS (FETCH YCOORD
                                                                                       OF (FETCH NODEPOSITION OF N)))
                                                                  DO ;; We won't match the other label node because it has a unique ypos
                                                                     (COMPARETEXT.SETSEL (COMPARETEXT.TSTREAM
                                                                                          N WINDOW (NOT INCOL1))
                                                                           N)
                                                                     (RETURN N)))
                                    (COMPARETEXT.SETSEL (COMPARETEXT.TSTREAM NODE WINDOW INCOL1)
                                           NODE)
               ELSE ;; The column header, set up the file window with no selection.
                    (COMPARETEXT.TSTREAM NODE WINDOW INCOL1)))))])
```

(IMCOMPARE.MIDDLEBUTTONFN

```
[LAMBDA (NODE WINDOW)                                                 ; Edited 27-Dec-2021 11:59 by rmk
                                                                      ; Edited 25-Dec-2021 11:51 by rmk
                                                                      ; Edited 24-Dec-2021 10:42 by rmk
                                                                      ; Edited 22-Dec-2021 16:08 by rmk
  ;; Edited 16-Dec-2021 10:55 by rmk: Remove previous HASH.TYPE from the middle mouse menu
                                                                      ; Edited 16-Dec-2021 10:51 by rmk
                                                                      (* mjs " 6-Jan-84 11:37")
  ;; This function is called if the MIDDLE mouse button is pressed over a graph node.  The selected node is IMCOMPARE-ed with the last node
  ;; selected <which is boxed>.  The type of hashing used <PARA, LINE, or WORD> is selected from a pop-up menu.  If none of the hashing types is
  ;; selected, the current node is boxed.  The pop-up menu is always located a little above the current cursor position, so a quick
  ;; double-MIDDLE-click is an easy way to change the current boxed node.
  (CL:WHEN NODE
      [PROG (INNER.HASH.TYPE REGION (LASTNODES (WINDOWPROP WINDOW 'LASTNODES))
                (PWINDOW (GETPROMPTWINDOW WINDOW)))
            (CLEARW PWINDOW)
            (CL:UNLESS LASTNODES
                (PRIN3 "Select nodes to be expanded" PWINDOW)
                (RETURN))
            [SETQ INNER.HASH.TYPE (MENU (create MENU
                                                TITLE _ "New hash type?"
                                                ITEMS _ (REMOVE (GRAPHERPROP (WINDOWPROP WINDOW 'GRAPH)
                                                                             'HASH.TYPE)
                                                                '(PARA LINE WORD))
                                                MENUOFFSET _ (create POSITION
                                                                     XCOORD _ 20
                                                                     YCOORD _ -20]
            (printout PWINDOW "Comparing chunks by " INNER.HASH.TYPE T)
           ;; Offset the region a little bit, so that the parent region is visible
            [SETQ REGION (COPY (WINDOWPROP WINDOW 'REGION]
            (ADD (FETCH (REGION LEFT) OF REGION)
                 30)
            (ADD (FETCH (REGION BOTTOM) OF REGION)
                 -30)
            (IMCOMPARE.CHUNKS (FETCH (GRAPHNODE NODEID) OF (CAR LASTNODES))
                   (FETCH (GRAPHNODE NODEID) OF (CADR LASTNODES))
                   INNER.HASH.TYPE REGION (CDR (GRAPHERPROP (WINDOWPROP WINDOW 'GRAPH)
                                                            'FILELABELS]))])
```

(IMCOMPARE.COPYBUTTONFN

```
[LAMBDA (WINDOW NODE)                                                 ; Edited 25-Dec-2021 13:26 by rmk
                                                                      ;
  ;; The grapher calls this with the window but not the node. So there must be some internal grapher stuff to find the node from the mouse
  ;; coordinates.  The goal would be to at least do a COPYINSERT of the filename.
```

```
    (HELP])

)

(FILESLOAD (SYSLOAD)
       GRAPHER REGIONMANAGER)

(DEFINEQ

(TAIL1
  [LAMBDA (ALL)                                          ; Edited 25-Dec-2021 21:54 by rmk
    (FOR X IN (CL:IF ALL
                  CHUNKLIST1
                  C1TAIL)
       COLLECT (LIST (FETCH FILEPTR OF X)
                     (FETCH FILEPTR OF (CAR (FETCH OTHERCHUNK OF X])


(TAIL2
  [LAMBDA (ALL)                                          ; Edited 25-Dec-2021 21:29 by rmk
    (FOR X IN (CL:IF ALL
                  CHUNKLIST2
                  C2TAIL)
       COLLECT (LIST (FETCH FILEPTR OF X)
                     (FETCH OTHERCHUNK OF X])

)

;; Debugging

(RPAQ? COMPARETEXT.ALLCHUNKS T)

(RPAQ? COMPARETEXT.AUTOTEDIT T)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(RECORD IMCOMPARE.CHUNK (HASHVALUE FILEPTR CHUNKLENGTH FILENAME . OTHERCHUNK)
       FILEPTR _ 1)

(RECORD IMCOMPARE.SYMB (NEWCOUNT OLDCOUNT . OLDPTR))
)

(FILESLOAD (LOADCOMP)
       GRAPHER)
)
```

FUNCTION INDEX

VARIABLE INDEX

RECORD INDEX