NAME

trv − Trace visualizer for FORK program trace files

SYNOPSIS

trv [*options ...*] [*input_file ...*] [*output_file ...*]

DESCRIPTION

trv reads tracing information from the *input file ,* and writes a trace graphic in **.fig −**format to the *output file* .

If *input file* is omitted, *trv* assumes **trace.trv** to be the input file, if *output file* is omitted, its filename is supposed to be **trace.fig .** If the input filename is given as **∗.trv ,** the output filename is **∗.fig .** Though it is recommended to use files with suffix **.trv** as trace files and files with **.fig** -suffixes as output, this convention is not mandatory.

−x *x_size*
scales vertical size of the output graphic in per one. Default value is 1.0 .

−r *ratio*
scales horizontal size of output graphic relative to *x_size. ratio* is computed after *x_size.* Default value is 1.5 (approx. DIN A4).

−w *bar_width*
determines width of trace bars (in pixels). Default is 240.

−c *config_file*
specifies an additional *CONFIGURATION* file which determines the appeareance of the graphic, overriding the settings in **trace.h .** For further information on CONFIGURATION read below.

−n *nostats option:* suppresses output of statistics on shloads, shstores, mpadds, mpmax's, mpands and mpors. Useful if text lines reach above page boundaries.

CONFIGURATIONS

Using *configurations,* you can define additional event types. Also is it possible to assign a certain *color* or *pattern* to an *event type.* You can assign color and width to the *MPI message arrows* and you can specify a certain *color range* to redefine the private color map. To do so, you have to create a *config file* and place it in your current working directory. Following is a description of the config file syntax.

NOTE: Any settings given in the config file override the according setting in the **trace.h** file. It is possible to add settings to the default configuration. However, changing the settings in the **trace.h** file is not recommendable.

CONFIGURATION SYNTAX

format TRACE_EVENT:
TRACE_EVENT(<traceEventNr>,<traceEventNameString>, <pen_color>, <fill_color>, <y_size>, <y_offset>, <fill_pattern>)

traceEventNr:
Enumerative identifier value for events. Can range from 0 to 31. Values from 0 to 11 are used by the standard configuration in **trace.h.**

NOTE:

The last event MUST BE a dummy event of the following form: TRACE_EVENT(x, TRACE_MAX_EVENT, ...) where **x** is the least unused event number (e.g. if you specify 12 events with event numbers ranging from 0 to 11, the last event must be specified like: TRACE_EVENT(12,TRACE_MAX_EVENT, ...)

traceEventNameString:
You can give identifier names to each event. Identifier names must be necessarily distinct from each other. By convention, they are written in capital letters, using undercores "_" as blanks.

pen_color: Border color value assigned to each event. Ranges from 0 to 31 (regular color palette of **XFIG). -1 as parameter says that**

Values: 0: black 1: blue 2: green 3: cyan 4: red 5: magenta 6: yellow

Just to name a few ones. For further information, consult the xfig manual.

fill_color: Fill color value assigned to each event. Similar to pen_color.

y_size: the width of the trace bars associated with the event. Default is 1.0

y_offset: offset, by which the trace bar is shifted down. Default is 0.0

pattern: specifies a pattern for the event. The pattern is drawn in the pen color. The value for no fill is 0

format TRACE_ARROW:
TRACE_ARROW(<inner_width>,<inner_color>, <outer_width>, <outer_color>)

inner_width:
width of inner arrow (in pixels)

inner_color:
color value of inner arrow

outer_width:
width of outer arrow (in pixels)

outer_color:
color value of outer arrow

format COLOR_RANGE:
COLOR_RANGE(<red_min>, <red_max>, <green_min>, <green_max>, <blue_min>, <blue_max>, <skew_power>)

The custom color table in xfig is filled with color values from 0 to 192. Each color consists of a red, a green and a blue fraction. With *COLOR_RANGE* you can assign the left value to 0 and the right value to 192; values in bitween are interpolated. With the skew parameter one can adjust the order of the interpolation (e.g. 2.0 for square, 3.0 for cubic) . Default is linear interpolation, which means 1.0 for the skew parameter.

red_min: start value for red color fraction

red_max: start value for red color fraction

green_min: start value for green color fraction

green_max:
start value for green color fraction

blue_min: start value for blue color fraction *blue_max:* start value for blue color fraction

CONFIG FILE EXAMPLE:

(taken from **trace.h**)

```
 TRACE_EVENT(0, TRACE_END, -1, -1, 1.0, 0.0, 20)
 TRACE_EVENT(1, TRACE_SPLIT, 5, 5, 1.0, 0.0, 20)
 TRACE_EVENT(2, TRACE_GROUP, -1, -1, 1.0, 0.0, 20)

/* ... */

 TRACE_EVENT(10,TRACE_ENTER_RECV, -1, -1, 0.5, 0.5, 20)
 TRACE_EVENT(11,TRACE_EXIT_RECV,  -1, -1, 2.0, -1.0, 20)
 TRACE_EVENT(12,TRACE_MAX_EVENT,  -1, -1, 0.0,  0.0, 20)

/* These were the event definitions ... */

 TRACE_ARROW(1, 4, 2, 7)

/* ... and this is the arrow definition: *
 * inner width 1, inner color red        *
 * outer width 2, outer color white      */

 COLOR_RANGE(20, 20, 0, 100, 192, 100, 3.0)

 /* Here the color range definition */
```

Let's have a look at the exemplaric event definition:

TRACE_EVENT(10, TRACE_ENTER_RECV, -1, -1, 0.5 , 0.5, 20)

traceEventNr is 8, *traceEventNameString* is TRACE_ENTER_RECV, *pen_color* shall be adopted by the preceding event, *fill_color* either. *y_size* is 0.5, *y_offset* is 0.0 This means, that the event bar is only half wide than normal, and because the offset is 0.0, it starts at 50 % of the width of the preceding event. The result is a "lower half". The y_size and y_offset values remain for every following event until they are reset. According to this,

TRACE_EVENT(9, TRACE_EXIT_RECV, -1, -1, 2.0, -1.0, 20)

has for y_size 2.0 (double bar width) and for y_offset -1.0 (start the last "large" event bar top), so that after TRACE_EXIT_RECV the mode before the last TRACE_ENTER_SEND is restored.

fcc(1), *pramsim*(1), *xfig*(1)