

## PlotHandler.java

```
1 //*****
2 //
3 // File:    PlotHandler.java
4 // Package: ---
5 // Unit:    Class PlotHandler
6 //
7 //*****
8
9 import java.awt.BasicStroke;
10 import java.awt.Color;
11 import java.io.IOException;
12 import java.text.DecimalFormat;
13
14 import edu.rit.numeric.ListXYSeries;
15 import edu.rit.numeric.plot.Dots;
16 import edu.rit.numeric.plot.Plot;
17
18 /**
19  * Class PlotHandler is the delegate for dealing with visualizing the data
20  * generated by the "number crunching" program, MrPotatoHead.
21  * Its purpose is to be instantiated in MrPotatoHead with the data to plot,
22  * where the write() method should then be called.
23  * <P>
24  * Running this program and specifying in the command line arguments the plot
25  * files previously generated will open a graphical representation of these
26  * plots for each file.
27  * </P>
28  * @author Jimi Ford
29  * @version 5-6-2015
30  *
31  */
32 public class PlotHandler {
33
34     // private data members
35     private final String rtTotalFile;
36     private final String dfTotalFile;
37     private final String rtlargeFile;
38     private final String dflargeFile;
39     private final String rtSmallFile;
40     private final String dfSmallFile;
41     private final String routerDropFile;
42     private final String reRouteFile;
43     private final String primaryActivityFile;
44     private final String secondaryActivityFile;
45     private final ListXYSeries dfTotal;
46     private final ListXYSeries rtTotal;
47     private final ListXYSeries dfLarge;
48     private final ListXYSeries rtlarge;
49     private final ListXYSeries dfSmall;
50     private final ListXYSeries rtSmall;
51     private final ListXYSeries aDrop;
52     private final ListXYSeries bDrop;
53     private final ListXYSeries cDrop;
54     private final ListXYSeries dDrop;
55     private final ListXYSeries aReRoute;
56     private final ListXYSeries bReRoute;
57     private final ListXYSeries cReRoute;
58     private final ListXYSeries dReRoute;
```

```

59 private final ListXYSeries adActivity;
60 private final ListXYSeries bdActivity;
61 private final ListXYSeries cdActivity;
62 private final ListXYSeries d2Activity;
63 private final ListXYSeries abActivity;
64 private final ListXYSeries acActivity;
65 private final ListXYSeries baActivity;
66 private final ListXYSeries bcActivity;
67 private final ListXYSeries caActivity;
68 private final ListXYSeries cbActivity;
69 private final ListXYSeries daActivity;
70 private final ListXYSeries dbActivity;
71 private final ListXYSeries dcActivity;
72
73
74 /**
75  * Construct a new PlotHandler object
76  *
77  * @param prefix the prefix to use for saving the files
78  * @param dfTotal the xy-series that contains the drop fraction info
79  * @param rtTotal the xy-series that contains the response time info
80  * @param dfLarge series containing the drop fraction for large packets
81  * @param rtLarge series containing the response time for large packets
82  * @param dfSmall series containing the drop fraction for small packets
83  * @param rtSmall series containing the response time for small packets
84  * @param aDrop series containing the drop fraction of router a
85  * @param bDrop series containing the drop fraction of router b
86  * @param cDrop series containing the drop fraction of router c
87  * @param dDrop series containing the drop fraction of router d
88  * @param aReRoute series containing the re-route fraction of router a
89  * @param bReRoute series containing the re-route fraction of router b
90  * @param cReRoute series containing the re-route fraction of router c
91  * @param dReRoute series containing the re-route fraction of router d
92  * @param adActivity series containing activity fraction of link ad
93  * @param bdActivity series containing activity fraction of link bd
94  * @param cdActivity series containing activity fraction of link cd
95  * @param d2Activity series containing activity fraction of link d2
96  * @param abActivity series containing activity fraction of link ab
97  * @param acActivity series containing activity fraction of link ac
98  * @param baActivity series containing activity fraction of link ba
99  * @param bcActivity series containing activity fraction of link bc
100 * @param caActivity series containing activity fraction of link ca
101 * @param cbActivity series containing activity fraction of link cb
102 * @param daActivity series containing activity fraction of link da
103 * @param dbActivity series containing activity fraction of link db
104 * @param dcActivity series containing activity fraction of link dc
105 */
106 public PlotHandler(String prefix,
107     ListXYSeries dfTotal, ListXYSeries rtTotal,
108     ListXYSeries dfLarge, ListXYSeries rtLarge,
109     ListXYSeries dfSmall, ListXYSeries rtSmall,
110     ListXYSeries aDrop, ListXYSeries bDrop, ListXYSeries cDrop,
111     ListXYSeries dDrop, ListXYSeries aReRoute, ListXYSeries bReRoute,
112     ListXYSeries cReRoute, ListXYSeries dReRoute,
113     ListXYSeries adActivity, ListXYSeries bdActivity,
114     ListXYSeries cdActivity, ListXYSeries d2Activity,
115     ListXYSeries abActivity, ListXYSeries acActivity,
116     ListXYSeries baActivity, ListXYSeries bcActivity,

```

```

117     ListXYSeries caActivity, ListXYSeries cbActivity,
118     ListXYSeries daActivity, ListXYSeries dbActivity,
119     ListXYSeries dcActivity) {
120         rtTotalFile = prefix + "-traversal-time.dwg";
121         dfTotalFile = prefix + "-drop-fraction.dwg";
122         rtLargeFile = prefix + "-traversal-time-large.dwg";
123         rtSmallFile = prefix + "-traversal-time-small.dwg";
124         dfLargeFile = prefix + "-drop-fraction-large.dwg";
125         dfSmallFile = prefix + "-drop-fraction-small.dwg";
126         routerDropFile = prefix + "-router-drop-fraction.dwg";
127         reRouteFile = prefix + "-re-route-fraction.dwg";
128         primaryActivityFile = prefix + "-primary-link-activity-fraction.dwg";
129         secondaryActivityFile = prefix +
130             "-secondary-link-activity-fraction.dwg";
131         this.dfTotal = dfTotal;
132         this.rtTotal = rtTotal;
133         this.dfLarge = dfLarge;
134         this.rtLarge = rtLarge;
135         this.dfSmall = dfSmall;
136         this.rtSmall = rtSmall;
137         this.aDrop = aDrop;
138         this.bDrop = bDrop;
139         this.cDrop = cDrop;
140         this.dDrop = dDrop;
141         this.aReRoute = aReRoute;
142         this.bReRoute = bReRoute;
143         this.cReRoute = cReRoute;
144         this.dReRoute = dReRoute;
145         this.adActivity = adActivity;
146         this.bdActivity = bdActivity;
147         this.cdActivity = cdActivity;
148         this.d2Activity = d2Activity;
149         this.abActivity = abActivity;
150         this.acActivity = acActivity;
151         this.baActivity = baActivity;
152         this.bcActivity = bcActivity;
153         this.caActivity = caActivity;
154         this.cbActivity = cbActivity;
155         this.daActivity = daActivity;
156         this.dbActivity = dbActivity;
157         this.dcActivity = dcActivity;
158     }
159
160     /**
161     * Save the plot information into files and display the plots.
162     *
163     * @throws IOException if it can't write to the file specified
164     */
165     public void write() throws IOException {
166         write("Total", "0.0", dfTotal, dfTotalFile, rtTotal, rtTotalFile);
167         write("Large Pkt", "0.0", dfLarge, dfLargeFile, rtLarge, rtLargeFile);
168         write("Small Pkt", "0.00", dfSmall, dfSmallFile, rtSmall, rtSmallFile);
169         writeRouterDrop();
170         writeRouterReRoute();
171         writePrimaryLinkActivity();
172         writeSecondaryLinkActivity();
173     }
174

```

```

175
176 /**
177  * write the router drop fraction plot
178  * @throws IOException if it can't write to the file specified
179  */
180 private void writeRouterDrop() throws IOException {
181     Plot routerDropFraction = new Plot()
182         .plotTitle("Router Drop Fraction")
183         .xAxisTitle ("Mean arrival rate (pkt/sec)")
184         .yAxisTitle ("Drop fraction")
185         .yAxisStart (0.0)
186         .yAxisEnd (1.0)
187         .yAxisTickFormat (new DecimalFormat ("0.0"))
188         .seriesDots(null)
189         .seriesColor(Color.RED)
190         .xySeries(aDrop)
191         .seriesColor(Color.ORANGE)
192         .seriesDots(Dots.circle(Color.ORANGE, new BasicStroke(),
193             Color.ORANGE, 7))
194         .xySeries(bDrop)
195         .seriesDots(null)
196         .seriesColor(Color.GREEN)
197         .xySeries(cDrop)
198         .seriesColor(Color.BLUE)
199         .xySeries(dDrop)
200         .labelColor(Color.RED)
201         .label("<b>A</b>", 42.5, .85)
202         .labelColor(Color.ORANGE)
203         .label("<b>B</b>", 42.5, .75)
204         .labelColor(Color.GREEN)
205         .label("<b>C</b>", 42.5, .65)
206         .labelColor(Color.BLUE)
207         .label("<b>D</b>", 42.5, .55);
208     Plot.write(routerDropFraction, routerDropFile);
209 }
210
211 /**
212  * write the primary link activity plot
213  * @throws IOException if it can't write to the file specified
214  */
215 private void writePrimaryLinkActivity() throws IOException {
216     Plot linkActivity = new Plot()
217         .plotTitle("Primary Link Activity")
218         .xAxisTitle ("Mean arrival rate (pkt/sec)")
219         .yAxisTitle ("Link Activity Fraction")
220         .yAxisStart (0.0)
221         .yAxisEnd (1.0)
222         .yAxisTickFormat (new DecimalFormat ("0.0"))
223         .seriesDots(null)
224         .seriesColor(Color.RED)
225         .xySeries(adActivity)
226         .seriesColor(Color.ORANGE)
227         .seriesDots(Dots.circle(Color.ORANGE, new BasicStroke(),
228             Color.ORANGE, 7))
229         .xySeries(bdActivity)
230         .seriesDots(null)
231         .seriesColor(Color.GREEN)
232         .xySeries(cdActivity)

```

```

233     .seriesColor(Color.BLUE)
234     .xySeries(d2Activity)
235     .labelColor(Color.RED)
236     .label("<b>A</b>", 42.5, .65)
237     .labelColor(Color.ORANGE)
238     .label("<b>B</b>", 42.5, .55)
239     .labelColor(Color.GREEN)
240     .label("<b>C</b>", 42.5, .45)
241     .labelColor(Color.BLUE)
242     .label("<b>D</b>", 42.5, .35);
243     Plot.write(linkActivity, primaryActivityFile);
244 }
245
246 /**
247  * write the secondary link activity plot
248  * @throws IOException if it can't write to the file specified
249  */
250 private void writeSecondaryLinkActivity() throws IOException {
251     Plot linkActivity = new Plot()
252     .plotTitle("Secondary Link Activity")
253     .xAxisTitle ("Mean arrival rate (pkt/sec)")
254     .yAxisTitle ("Link Activity Fraction")
255     .yAxisStart (0.0)
256     .yAxisEnd (1.0)
257     .yAxisTickFormat (new DecimalFormat ("0.0"))
258     .seriesDots(null)
259     .seriesColor(Color.RED)
260     .xySeries(abActivity)
261     .xySeries(acActivity)
262     .seriesColor(Color.ORANGE)
263     .seriesDots(Dots.circle(Color.ORANGE, new BasicStroke(),
264         Color.ORANGE, 7))
265     .xySeries(bcActivity)
266     .xySeries(baActivity)
267     .seriesColor(Color.GREEN)
268     .seriesDots(null)
269     .xySeries(caActivity)
270     .xySeries(cbActivity)
271     .seriesColor(Color.BLUE)
272     .xySeries(daActivity)
273     .xySeries(dbActivity)
274     .xySeries(dcActivity)
275     .labelColor(Color.RED)
276     .label("<b>A</b>", 42.5, .45)
277     .labelColor(Color.ORANGE)
278     .label("<b>B</b>", 42.5, .35)
279     .labelColor(Color.GREEN)
280     .label("<b>C</b>", 42.5, .25)
281     .labelColor(Color.BLUE)
282     .label("<b>D</b>", 42.5, .15);
283     Plot.write(linkActivity, secondaryActivityFile);
284 }
285
286 /**
287  * write the router re-route fraction plot
288  *
289  * @throws IOException if it can't write to the file specified
290  */

```

```

291 private void writeRouterReRoute() throws IOException {
292     Plot reRouteFraction = new Plot()
293         .plotTitle("Router Re-Route Fraction")
294         .xAxisTitle ("Mean arrival rate (pkt/sec)")
295         .yAxisTitle ("Re-Route fraction")
296         .yAxisStart (0.0)
297         .yAxisEnd (1.0)
298         .yAxisTickFormat (new DecimalFormat ("0.0"))
299         .seriesDots(null)
300         .seriesColor(Color.RED)
301         .xySeries(aReRoute)
302         .seriesColor(Color.ORANGE)
303         .seriesDots(Dots.circle(Color.ORANGE, new BasicStroke(),
304             Color.ORANGE, 7))
305         .xySeries(bReRoute)
306         .seriesDots(null)
307         .seriesColor(Color.GREEN)
308         .xySeries(cReRoute)
309         .seriesColor(Color.BLUE)
310         .xySeries(dReRoute)
311         .labelColor(Color.RED)
312         .label("<b>A</b>", 42.5, .55)
313         .labelColor(Color.ORANGE)
314         .label("<b>B</b>", 42.5, .45)
315         .labelColor(Color.GREEN)
316         .label("<b>C</b>", 42.5, .35)
317         .labelColor(Color.BLUE)
318         .label("<b>D</b>", 42.5, .25);
319     Plot.write(reRouteFraction, reRouteFile);
320 }
321
322 /**
323  * Save the plot information into files.
324  *
325  * @param titlePrefix Prefix of the plot's title
326  * @param yFormat decimal format of the traversal time y-axis labels
327  * @param df drop fraction series
328  * @param dfFile drop fraction file name
329  * @param rt response time series
330  * @param rtFile response time file
331  * @throws IOException if it fails to write to any of the specified files
332  */
333 private void write(String titlePrefix, String yFormat, ListXYSeries df,
334     String dfFile, ListXYSeries rt, String rtFile) throws IOException {
335     Plot responseTime = new Plot()
336         .plotTitle (titlePrefix+" Traversal Time")
337         .xAxisTitle ("Mean arrival rate (pkt/sec)")
338         .yAxisTitle ("Mean traversal time (sec)")
339         .yAxisTickFormat (new DecimalFormat (yFormat))
340         .seriesDots (null)
341         .xySeries (rt);
342     Plot dropFraction = new Plot()
343         .plotTitle (titlePrefix+" Drop Fraction")
344         .xAxisTitle ("Mean arrival rate (pkt/sec)")
345         .yAxisTitle ("Drop fraction")
346         .yAxisStart (0.0)
347         .yAxisEnd (1.0)
348         .yAxisTickFormat (new DecimalFormat ("0.0"))

```

```

349     .seriesDots (null)
350     .xySeries (df);
351     Plot.write(responseTime, rtFile);
352     Plot.write(dropFraction, dfFile);
353 }
354
355 /**
356  * Open a GUI for each plot in order to visualize the results of a
357  * previously run set of simulations.
358  *
359  * @param args each plot file generated that you wish to visualize
360  */
361 public static void main(String args[]) {
362     if(args.length < 1) {
363         System.err.println("Must specify at least 1 plot file.");
364         usage();
365     }
366
367     for(int i = 0; i < args.length; i++) {
368         try {
369             Plot plot = Plot.read(args[i]);
370             plot.getFrame().setVisible(true);
371         } catch (ClassNotFoundException e) {
372             System.err.println("Could not deserialize " + args[i]);
373         } catch (IOException e) {
374             System.err.println("Could not open " + args[i]);
375         } catch (IllegalArgumentException e) {
376             System.err.println("Error in file " + args[i]);
377         }
378     }
379 }
380
381 /**
382  * Print the usage message for this program and gracefully exit.
383  */
384 private static void usage() {
385     System.err.println("usage: java PlotHandler <plot-file-1> "+
386         "<(<plot-file-2> <plot-file-3>... etc.)");
387     System.exit(1);
388 }
389 }
390

```