Example and comparison of path distance estimation

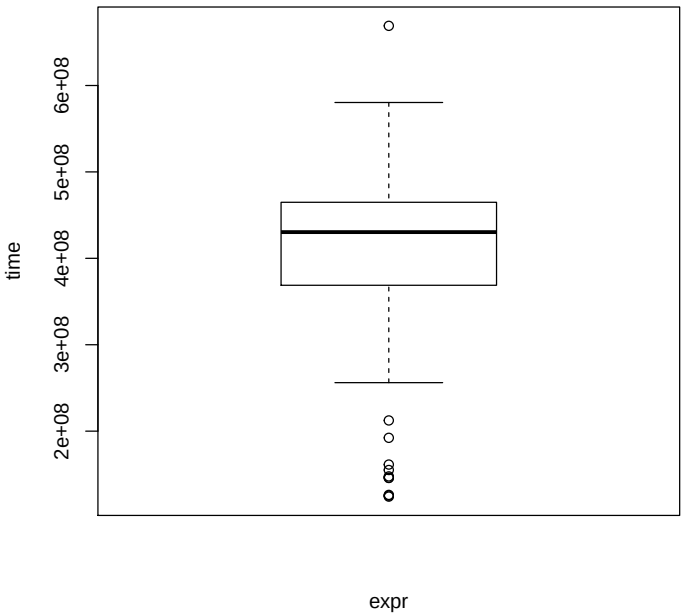
```
> library(networkDynamicData)
> library(tsna)
> library(microbenchmark)
> data(concurrencyComparisonNets)
```

1 paths.fwd.earliest (Dijkstra DFS)

How long does it take for networks we are interested in?

Evaluate earliest fwd path from 100 random vertices in the 'base' network.

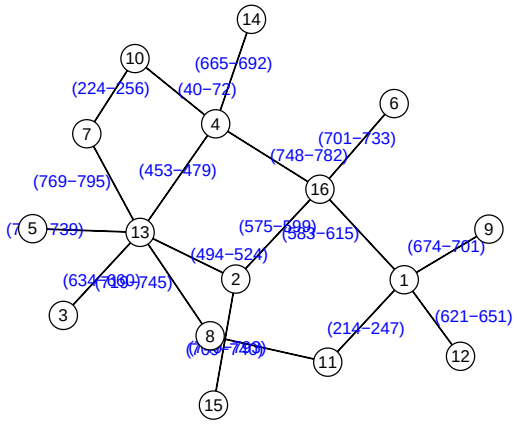
```
> network.size(base)

[1] 1000

> network.edgecount(base)

[1] 1916

> #
> times<-microbenchmark(paths.fwd.earliest(base,
+                v=round(runif(1,min=1,max=network.size(base)))))
> print(times,unit='s')

Unit: seconds
                                                                                 expr
 paths.fwd.earliest(base, v = round(runif(1, min = 1, max = network.size(base))))
       min        lq    median        uq       max neval
 0.1242559 0.3688037 0.4303218 0.4648538 0.6690303   100

> plot(times)
```



So a little less than half a second for each forward-set

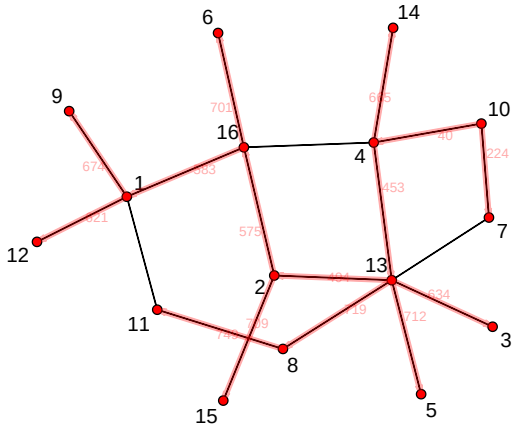
Hilite possible paths found on Moody's example. Plot a view of network with edge and vertex labels

```
> data(moodyContactSim)
> plot(moodyContactSim,
+      displaylabels=TRUE,
+      label.cex=0.8,
+      label.pos=5,
+      vertex.col='white',
+      vertex.cex=3,
+      edge.label=sapply(get.edge.activity(moodyContactSim),function(e){
+        paste('(',e[,1],'-',e[,2],')',sep='')
+      }),
+      edge.label.col='blue',
+      edge.label.cex=0.8
+    )
```



Extract the path, and plot

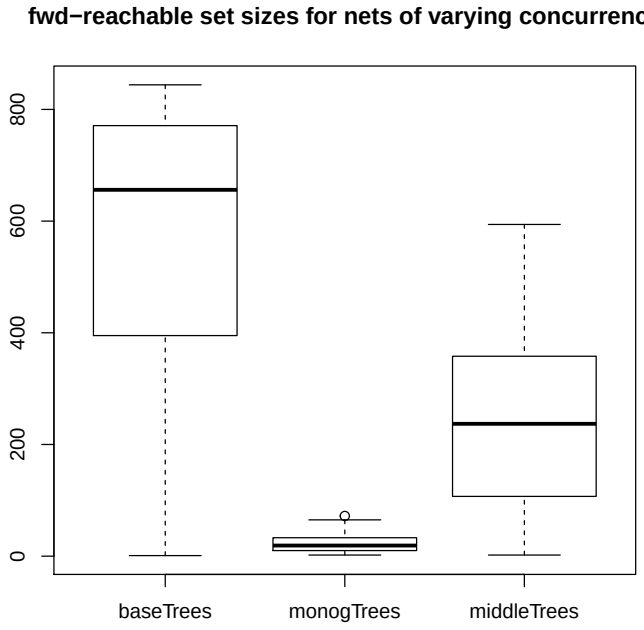
```
> v10path<-paths.fwd.earliest(moodyContactSim,v=10,start=0)
> plotpath(moodyContactSim,v10path,displaylabels=TRUE)
```



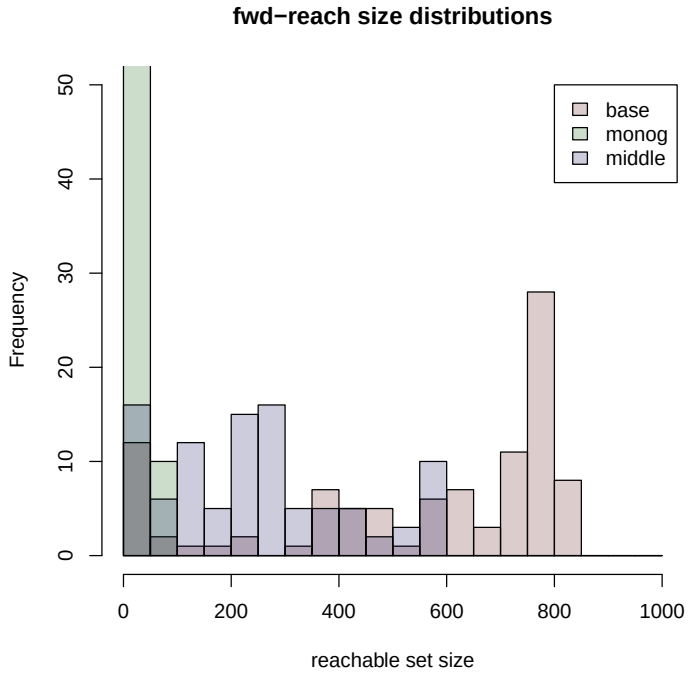
2 Compare earliest forward set sizes

An attempt to characterize networks by looking at the distribution of sizes of fwd reachable sets. All three networks have the same size, relationship duration distribution and cross-sectional mean degree, but different cross-sectional degree distributions.

```
> data(concurrencyComparisonNets)
> baseTrees<-reachable_set_sizes(base,sample=100)
> monogTrees<-reachable_set_sizes(monog,sample=100)
> middleTrees<-reachable_set_sizes(middle,sample=100)
> boxplot(cbind(baseTrees,monogTrees,middleTrees),
+         main='fwd-reachable set sizes for nets of varying concurrency')
```




```
> hist(baseTrees, main='fwd-reach size distributions',
+      ylim=c(0,50),xlim=c(0,1000),
+      breaks=seq(from=0,to=1000,by=50),
+      col='#55000033',xlab='reachable set size')
> hist(monogTrees,ylim=c(0,50),xlim=c(0,1000),
+      breaks=seq(from=0,to=1000,by=50),
+      col='#00550033',add=TRUE)
> hist(middleTrees,ylim=c(0,50),xlim=c(0,1000),
+      breaks=seq(from=0,to=1000,by=50),
+      col='#00005533',add=TRUE)
> legend(800,50,legend=c('base','monog','middle'),
+        fill=c('#55000033','#00550033','#00005533'))
```

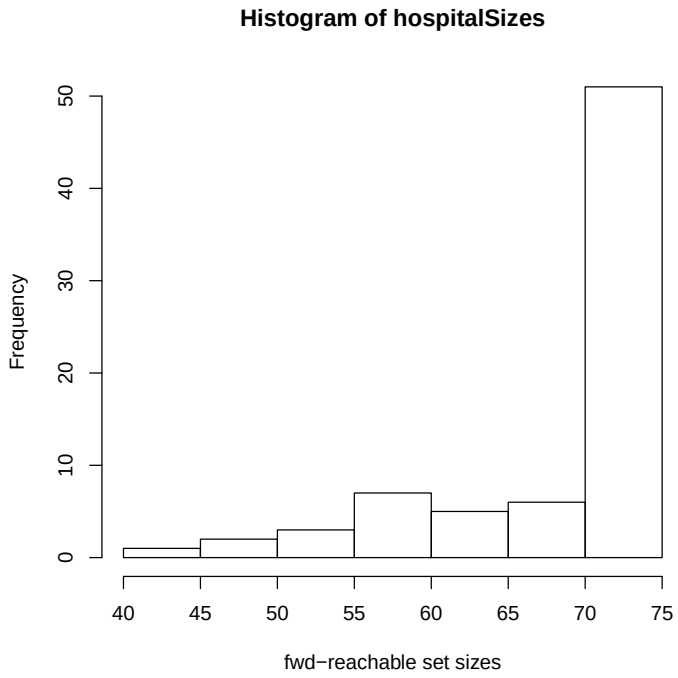


Q/TODO: are these distributions different from what we get from just plotting momentary degree distributions?

3 Hospital contacts

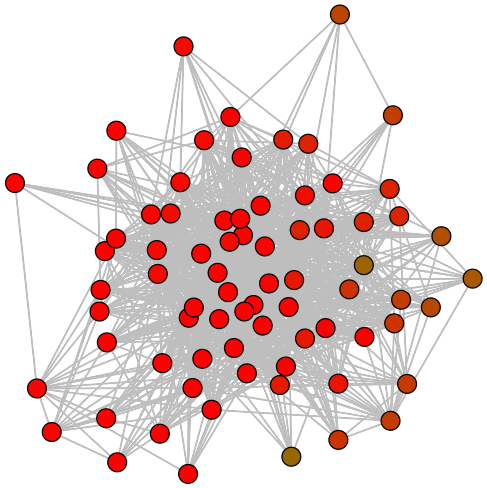
What does it look like for a realworld transmission network? Hospital contact is several days of RFID badge proximities in a French hospital ward.

```
> data(hospital_contact)
> network.size(hospital)

[1] 75

> network.edgecount(hospital)

[1] 1139

> hospitalSizes<-reachable_set_sizes(hospital)
> hist(hospitalSizes,xlab='fwd-reachable set sizes')
```



Plot the time-aggregated network, with vertex redness corresponding to size of reachable net originating at that vertex.

```
> plot(hospital,edge.col='gray',
+      vertex.col=rgb(hospitalSizes/max(hospitalSizes),
+                     1-hospitalSizes/max(hospitalSizes),0),
+      vertex.cex=2)
>
```



Perhaps the people with low connectivity are late arrivials?

4 density measures

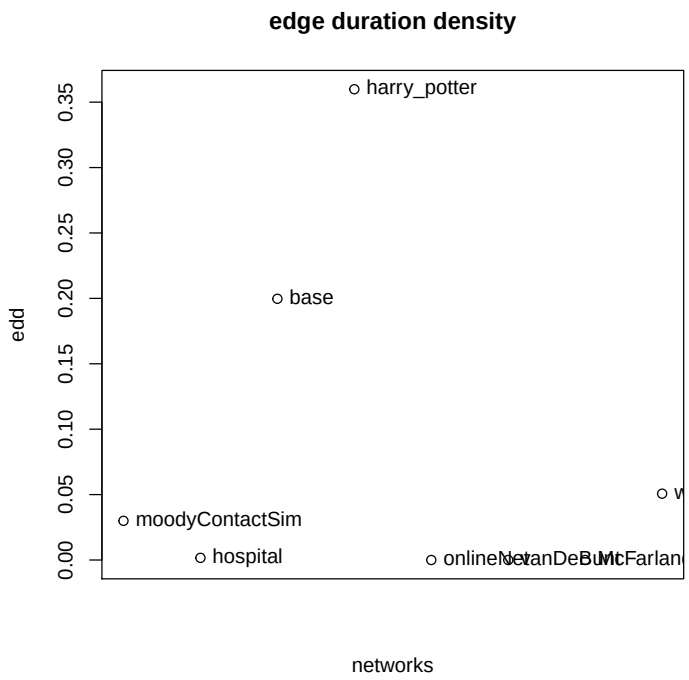
compare the values of the various density measures on various example networks

```
> data(moodyContactSim)
> data(harry_potter_support)
> data(onlineNetwork)
> data(vanDeBunt_students)
> data(McFarland_cls33_10_16_96)
> data(windsurfers)
> nets<-list(
+   moodyContactSim=moodyContactSim,
+   hospital=hospital,
+   base=base,
+   harry_potter=harry_potter_support,
+   onlineNet=onlineNet,
+   vanDeBunt=vanDeBunt_students,
+   McFarland=cls33_10_16_96,
+   windsurfers=windsurfers
+ )
```

>

What fraction of the edges are active at any time point?

```
> edd<-sapply(nets,edge_duration_density)
> plot(edd,main='edge duration density',xaxt='n',xlab='networks')
> text(edd,label=names(edd),pos=4)
```

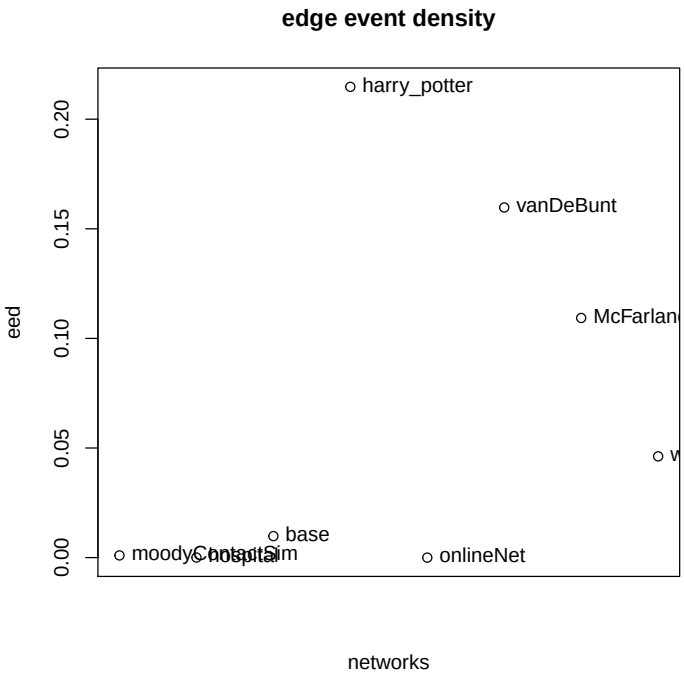


Notice that onlineNet, vanDeBunt and McFarland's classroom all have momentary events, so durations are not a very useful metric. Hospital has very short durations (20 seconds) compared to the overall time (350,000 seconds). Really should be corrected to measure as if it was discrete time with 20 sec timesteps.

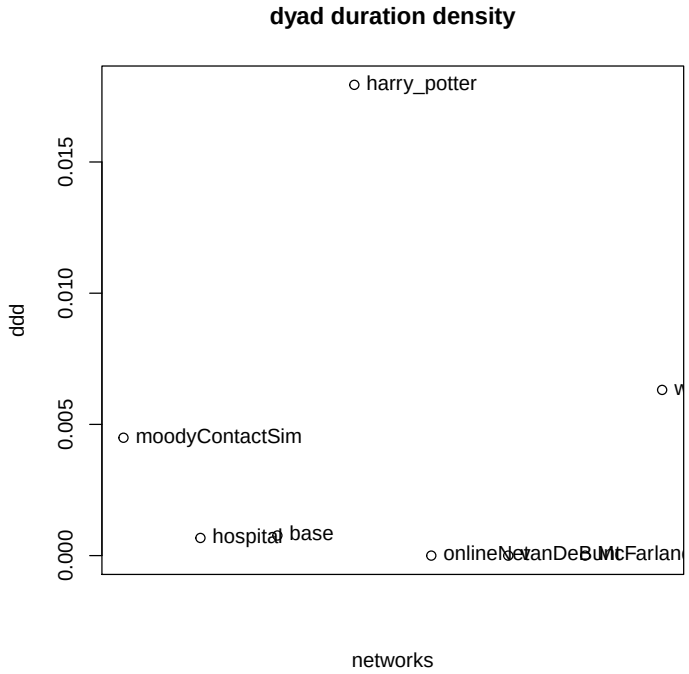
If we look at it by events, the momentary event networks rank much higher.

How many events are there in a unit of time? (of course this depends on having sensible units of time)

```
> eed<-sapply(nets,edge_event_density)
> plot(eed,main='edge event density',xaxt='n',xlab='networks')
> text(eed,label=names(eed),pos=4)
```




```
> ddd<-sapply(nets,dyad_duration_density)
> plot(ddd,main='dyad duration density',xaxt='n',xlab='networks')
> text(ddd,label=names(ddd),pos=4)
```



Windsurfers should probably have a correction for dyad duration density, since the vertex set varies, so not not all dyads are always availible.

The problem with these measures is that, like density, the values will be very, very low for real world networks so kind of hard to compare.

5 session info

```
> sessionInfo()

R version 3.0.3 (2014-03-06)
Platform: i686-pc-linux-gnu (32-bit)

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base
```

```
other attached packages:
[1] microbenchmark_1.3-0   tsna_0.1               networkDynamicData_0.2
[4] networkDynamic_0.7     network_1.10

loaded via a namespace (and not attached):
[1] statnet.common_3.2-11882-11883.1-2013.02.23-18.06.22
[2] tools_3.0.3
```