The biGraph–package

`Version 0.9-3`

Ingo Vogt

Chemogenomics Laboratory, Research Unit on Biomedical Informatics, Municipal Institute of Medical Research (IMIM) and University Pompeu Fabra, Parc de Recerca Biomèdica, Doctor Aiguader 88, 08003 Barcelona, Catalonia, Spain

January 25, 2011

This package is an extension to the *igraph*–package [1] and provides a set of tools specifically aimed at the analysis of bipartite graphs. While intended to be continuously extended regarding functionality, the current focus lies on the projection of bipartite graphs, specifically concerning the associated information loss. In addition, clustering and community detection among vertex subsets is supported by providing metric distance calculations based on flexible (weighted) neighbourhoods.

1 Projection of bipartite graphs

In *bipartite* graphs, vertices are distinguished into *top* and *bottom* ($\top$ and $\bot$) set and edges only occur between these two sets.

As example we examine a very small bipartite graph where vertices resemble diseases and genes thought to be involved some of these diseases.

```
> library(biGraph)
> data(g)
> summary(g)

Vertices: 38
Edges: 48
Directed: FALSE
No graph attributes.
Vertex attributes: name, type, dclass, col, dname, size.
Edge attributes: weight.
```

The vertex attribute `type` was derived in advanced by using the *igraph*–function `is.bipartite`, and this logical attribute identifies both vertex subsets.

```
> V(g)[type == TRUE]

Vertex sequence:
 [1] "Papillary_serous_carcinoma"         "Ovarian_cancer"
 [3] "Breast_cancer"                      "Prostate_cancer"
 [5] "Pancreatic_cancer"                  "Wilms_tumor"
 [7] "Fanconi_anemia"                     "Lymphoma"
```

```
 [9] "Ataxia-telangiectasia"              "T-cell_lymphoblastic_leukemia"
[11] "Amyotrophic_lateral_sclerosis"      "Spastic_ataxia/paraplegia"
[13] "Spinal_muscular_atrophy"            "Perineal_hypospadias"
[15] "Androgen_insensitivity"             "Lipodystrophy"
[17] "Charcot-Marie-Tooth_disease"        "Silver_spastic_paraplegia_syndrome"
[19] "Sandhoff_disease"
```

```
> V(g)[type == FALSE]

Vertex sequence:
 [1] "BRCA1"  "BRCA2"  "TP53"   "MAD1L1" "PIK3CA" "MSH2"   "ATM"    "ALS2"
 [9] "VAPB"   "RAD54L" "AR"     "CDH1"   "BRIP1"  "LMNA"   "KRAS"   "CHEK2"
[17] "GARS"   "BSCL2"  "HEXB"
```

The depiction of this bipartite graph shown in Figure˜1 (p.˜3) can be generated by

```
> plot(g, layout = layout.fruchterman.reingold, vertex.label = gsub("_",
+     "\n", V(g)$dname), vertex.color = V(g)$col, vertex.label.color = "black",
+     vertex.label.dist = 0.5, vertex.label.family = "Helvetica",
+     vertex.label.cex = 0.6, edge.label.cex = 0.6)
```

Often, it is desired to derive a monopartite projection regarding one vertex subset, where vertices belonging to this selected set are connected by an edge if they are connected to at least one vertex of the other vertex subset.

For this purpose, *igraph* provides the function `bipartite.projection`, that generates both projections, from which the disease projection is shown in Figure˜2 (p.˜4)

```
> proj <- bipartite.projection(g)
> plot(proj[[2]], layout = layout.fruchterman.reingold, vertex.label = gsub("_",
+     "\n", V(g)[type == TRUE]$name), vertex.color = V(g)[type ==
+     TRUE]$col, vertex.label.color = "black", vertex.label.dist = 0.5,
+     vertex.label.family = "Helvetica", vertex.label.cex = 0.6,
+     edge.label.cex = 0.6)
```

However, this projection comes not without a loss of information as discussed in [2, 3, 4, **?**, 5]. In order to allow the quantification of this information loss two approaches are presented in [6], which are implemented in *biGraph* by function `bipartite.projection.informationloss`.

1.1 Assessing information loss

In *biGraph*, function `bipartite.projection.informationloss` generates the desired projection(s) and subsequently measures the information loss according to the user selection.

- `graph.dH`: Change in uncertainty for entire graph
- `vertex.dH`: Change in uncertainty for single vertices
- `edge.dH`: Change in uncertainty for single edges
- `covLoss`: Loss of coverage for single edges, leads to automatic calculation of vertex attribute `avCovLoss`

As a shortcut, `measures='complete'` selects all measures at once.

```
> proj <- bipartite.projection.informationloss(g, vType = TRUE,
+     measures = "complete")
```

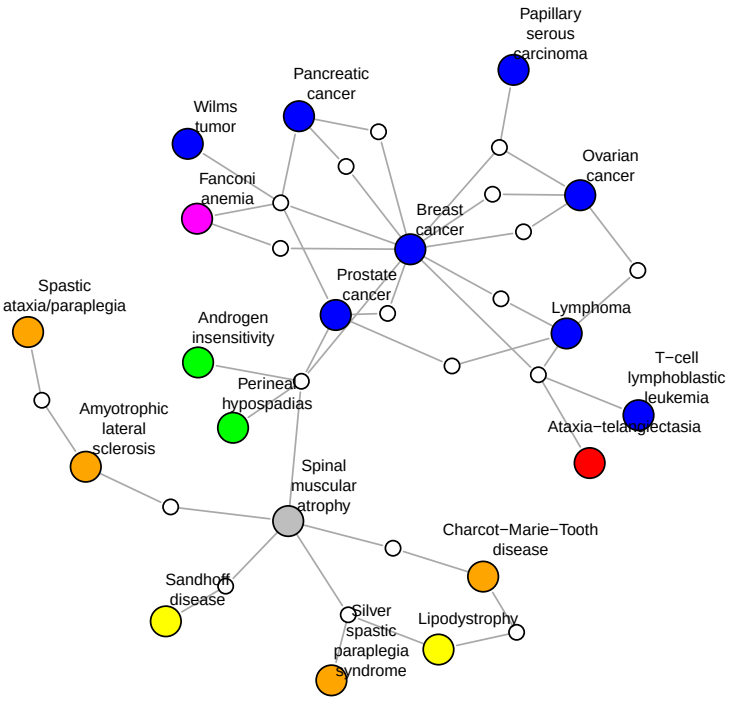

Figure 1: Bipartite gene–disease graph, names of genes (smaller white circles) are omitted for better readability

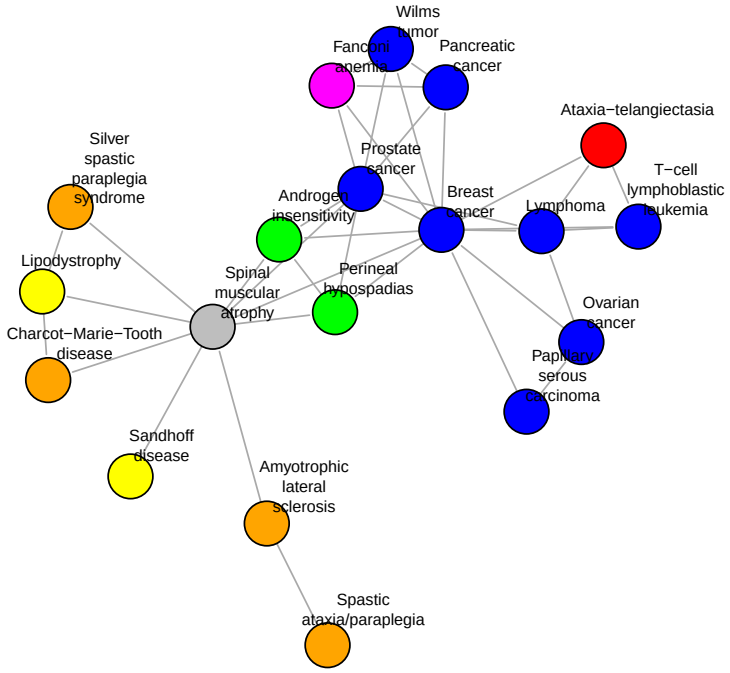

Figure 2: Monopartite projection for disease vertices

```
> proj[[1]]$H.delta

[1] 1.796607

> summary(V(proj[[1]])$H.delta)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.4055  0.9253  1.5260  1.5840  2.0790  2.7730

> summary(E(proj[[1]])$H.delta)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.6931  1.3860  1.3920  2.0790  2.0790

> summary(E(proj[[1]])$covLoss)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.2625  0.5000  0.3955  0.6000  0.6000

> summary(V(proj[[1]])$avCovLoss)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.1852  0.3407  0.3352  0.5000  0.6000
```

Figure˜3 (p.˜6) shows the monopartite disease projection where information loss measurements are visualized. If multiple cpus or cores are locally available, it is recommended to install the `multicore`–package and enable parallel computation via setting `parall=TRUE` and `cores` as desired. To facilitate the analysis of information loss e.g. by means of Figure˜4 (p.˜7), attributes `H.before` and `H.after` are also assigned to the corresponding graph entities when measuring the change in uncertainty.

```
> plot(V(proj[[1]])$H.before, V(proj[[1]])$H.after, xlab = "Uncertainty before projection [nats]",
+     ylab = "Uncertainty after projection [nats]", pch = 20, cex = 1.5,
+     xlim = c(0, 4), ylim = c(0, 4))
```

1.2 Generating weighted projections

If desired, one can generate a projection where graph entities are assigned a weight, whether it is based on some properties derived from the original bipartite graph or other sources. This functionality is implemented in function `get.weighted.projection`. Currently only edge weights are supported, with one built–in weighting scheme and the possibility to provide an externally calculated weight matrix.

Apart from taking the number of shared neighbours in the bipartite graph also the weighting scheme for collaboration networks introduced by Newman [7] has been implemented. In Figure˜5 (p.˜8) the disease projection is shown with the number of shared neighbours as edge weights.

```
> proj_weighted <- get.weighted.projection(g, vType = TRUE, mode = "shared-neighbours")
> plot(proj_weighted, layout = layout.fruchterman.reingold, vertex.color = V(g)[type ==
+     TRUE]$col, vertex.label = "", vertex.label.family = "Helvetica",
+     edge.label.family = "Helvetica", edge.label = E(proj_weighted)$weight,
+     vertex.label.cex = 0.6, edge.label.cex = 0.6, vertex.label.dist = 0.5)
```

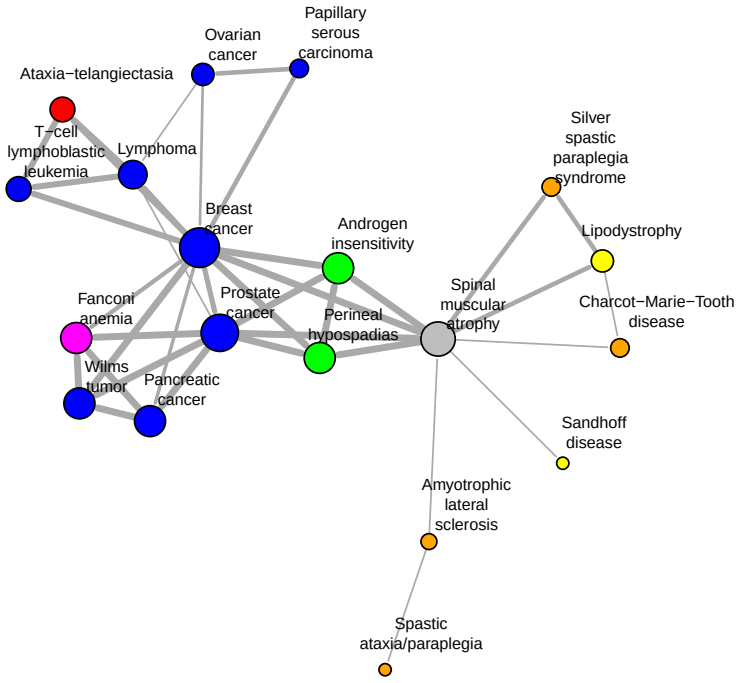

Figure 3: Monopartite disease projection. Vertex size and edge width are scaled by increase in uncertainty and loss of coverage, respectively.

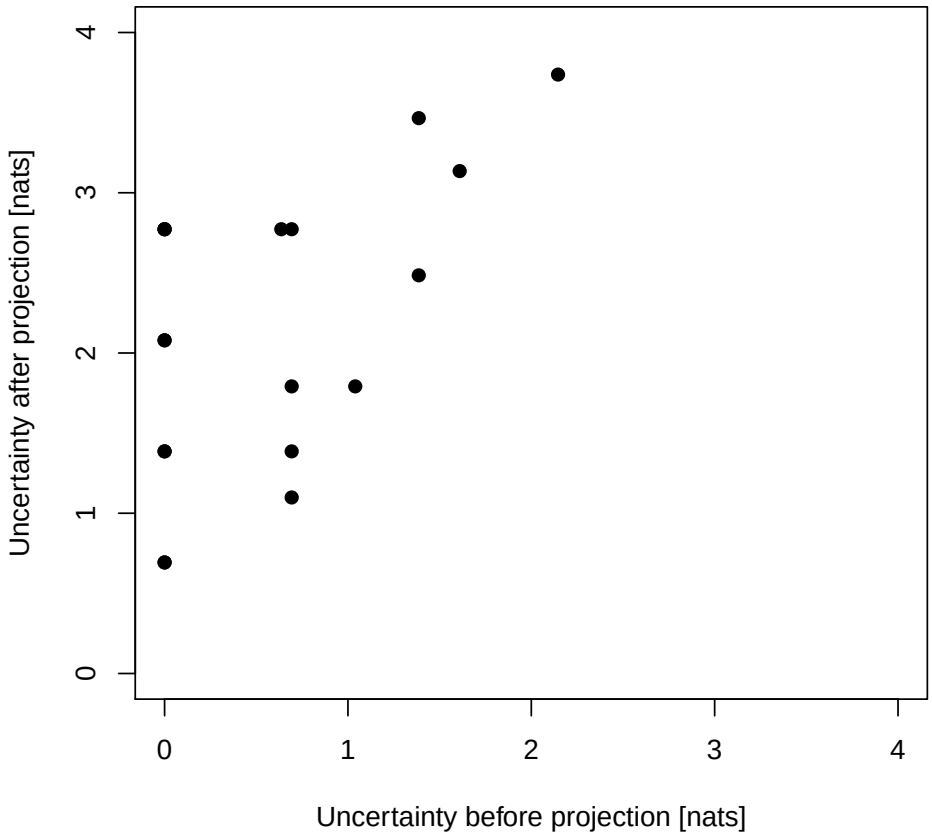

Figure 4: Comparison of uncertainty associated to vertices before and after projection

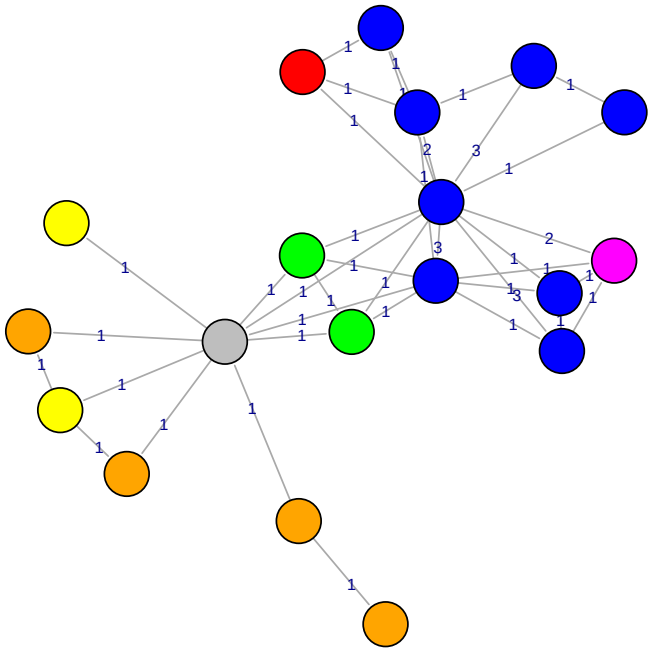

Figure 5: Disease projection with number of shared neighbours as edge weights

2 Linkage patterns

As one of the basic principles introduced in [6], linkage patterns in the context of projected monopartite graphs are defined as the different neighbourhoods of order 1 of vertices that are omitted during projection.

```
> lpi(g,vType=FALSE)

Linkage.Patterns
 [1] Breast_cancer;Ovarian_cancer;Papillary_serous_carcinoma
 [2] Breast_cancer;Fanconi_anemia;Pancreatic_cancer;Prostate_cancer;Wilms_tumor
 [3] Breast_cancer;Pancreatic_cancer
 [4] Lymphoma;Prostate_cancer
 [5] Breast_cancer;Ovarian_cancer
 [6] Lymphoma;Ovarian_cancer
 [7] Ataxia-telangiectasia;Breast_cancer;Lymphoma;T-cell_lymphoblastic_leukemia
 [8] Amyotrophic_lateral_sclerosis;Spastic_ataxia/paraplegia
 [9] Amyotrophic_lateral_sclerosis;Spinal_muscular_atrophy
[10] Breast_cancer;Lymphoma
[11] Androgen_insensitivity;Breast_cancer;Perineal_hypospadias;Prostate_cancer;Spinal_muscular_atrophy
[12] Breast_cancer;Fanconi_anemia
[13] Charcot-Marie-Tooth_disease;Lipodystrophy
[14] Breast_cancer;Prostate_cancer
[15] Charcot-Marie-Tooth_disease;Spinal_muscular_atrophy
[16] Lipodystrophy;Silver_spastic_paraplegia_syndrome;Spinal_muscular_atrophy
[17] Sandhoff_disease;Spinal_muscular_atrophy

Occurrences
 [1] 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1

Sources
 [1] BRCA1       BRCA2       KRAS;TP53   MAD1L1      CDH1;PIK3CA MSH2
 [7] ATM         ALS2        VAPB        RAD54L      AR          BRIP1
[13] LMNA        CHEK2       GARS        BSCL2       HEXB
```

Here, we observe 17 linkage patterns found among genes, of which pattern three and five occur twice, as mirrored by the corresponding source observations. There are three different modes for the generation of the linkage pattern information: `full` (default), `basic`, and `minimal`, where the last two increasingly discard the information on occurrences and sources. Via `write.lpi` the given linkage pattern information can be written to file, and read back via `read.lpi`.

```
> write.lpi(lpi(g, vType = FALSE, mode = "minimal"), "minimal.lpi")
> lp <- read.lpi("minimal.lpi")
> for (i in V(lp$graph)) {
+     c <- V(g)[which(V(g)$name == V(lp$graph)[i]$name) - 1]$col
+     if (length(c) > 0) {
+         V(lp$graph)[i]$col = c
+         V(lp$graph)[i]$size = 10
+     }
+     else {
+         V(lp$graph)[i]$col = "white"
+         V(lp$graph)[i]$size = 5
+     }
+ }
```

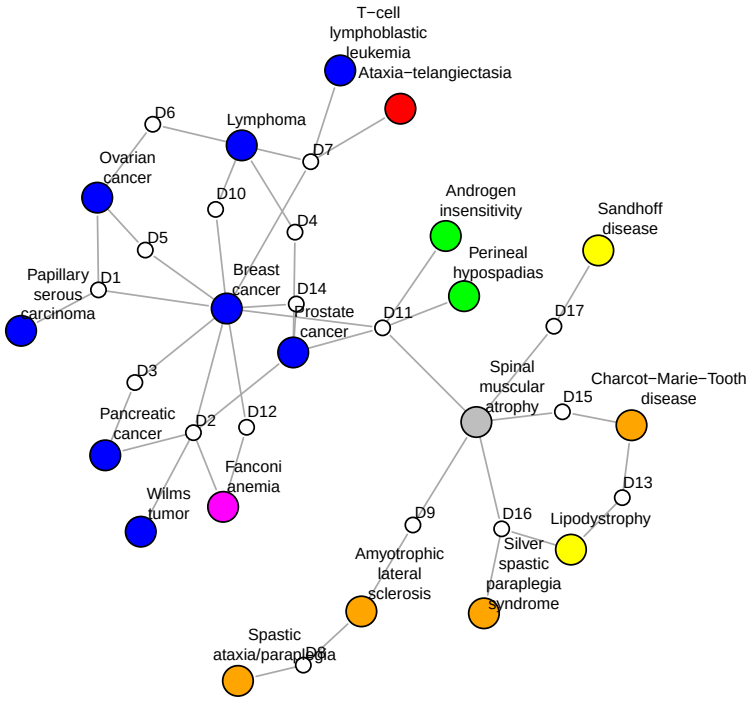

Figure 6: Reconstructed minimal bipartite graph.

```
> plot(lp$graph, layout = layout.fruchterman.reingold, vertex.label = gsub("_",
+     "\n", V(lp$graph)$name), vertex.label.family = "Helvetica",
+     vertex.label.cex = 0.6, edge.label.cex = 0.6, vertex.label.color = "black",
+     vertex.label.dist = 0.5, vertex.color = V(lp$graph)$col,
+     vertex.size = V(lp$graph)$size)
```

As can be seen in Figure˜6 (p.˜10), gene vertices have been replaced by vertices representing a linkage pattern each. If `mode='basic'` is chosen, only the names of omitted vertices will be discarded and the reconstructed graph and the original bipartite graph will be isomorphic. This could e.g. be useful if one would like to share the basic graph structure while withholding the identity of omitted vertices.

```
> write.lpi(lpi(g, vType = FALSE, mode = "basic"), "basic.lpi")
> lp <- read.lpi("basic.lpi")
> graph.isomorphic(g, lp$graph)

[1] TRUE
```

3 Analytical tools for bipartite graphs

Many tools that are used to study the properties of monopartite graphs cannot be directly applied to bipartite graphs. However, over the last decade, for some of these tools adaptions or analogous concepts have been devised, that nevertheless remain mostly unused and are usually missing from widely–used network and graph analysis suites. Apart from new developments, *biGraph* is also intended to collect those as well. In this first release we implemented some of these tools published by Borgatti and Everett [8], compassing measures for density, vertex centrality, and centralization with respect to each vertex subset.

3.1 Density

Density measures the number of edges present in a graph, usually as ratio of present over the number of all possible edges. *igraph* already provides a method to calculate the density of monopartite graphs, yet in bipartite graphs one has to take into account that vertices can only be connected to vertices in the other subset. In *biGraph* this is implemented in function `bipartite.graph.density`.

Density of bipartite graph according to function in *igraph*:

```
> graph.density(g)

[1] 0.0682788
```

Density of bipartite graph according to adapted function in *biGraph*:

```
> bipartite.graph.density(g)

$Density
[1] 0.132964
```

3.2 Vertex centrality

In graph and network analysis, vertex centrality measures are applied in order to determine the relative importance of vertices. The most widely used centrality measure are degree, closeness, betweenness, and eigenvector centrality. Currently, the adaption to bipartite graphs have been implemented for the first three measures. A visualization of the bipartite betweenness centrality scores is depicted in Figure˜7 (p.˜12).

```
> degree.centrality(g)

$Bipartite.Degree.Centrality
 [1] 0.15789474 0.05263158 0.21052632 0.57894737 0.26315789 0.21052632
 [7] 0.15789474 0.05263158 0.10526316 0.10526316 0.10526316 0.21052632
[13] 0.10526316 0.10526316 0.21052632 0.05263158 0.05263158 0.10526316
[19] 0.10526316 0.05263158 0.10526316 0.26315789 0.10526316 0.26315789
[25] 0.05263158 0.05263158 0.10526316 0.10526316 0.10526316 0.10526316
[31] 0.10526316 0.10526316 0.10526316 0.10526316 0.15789474 0.05263158
[37] 0.10526316 0.05263158
```

```
> bipartite.closeness.centrality(g)

$Bipartite.Closeness.Centrality
 [1] 0.4621849 0.3548387 0.3793103 0.6179775 0.4782609 0.4954955 0.3741497
 [8] 0.3642384 0.3691275 0.4471545 0.4135338 0.3900709 0.4545455 0.3293413
[15] 0.4782609 0.3642384 0.3642384 0.2763819 0.3333333 0.2340426 0.4135338
[22] 0.5339806 0.4621849 0.6043956 0.4330709 0.4330709 0.4545455 0.4471545
[29] 0.2791878 0.3333333 0.3293413 0.4471545 0.4545455 0.4074074 0.4135338
[36] 0.3254438 0.4014599 0.3179191
```

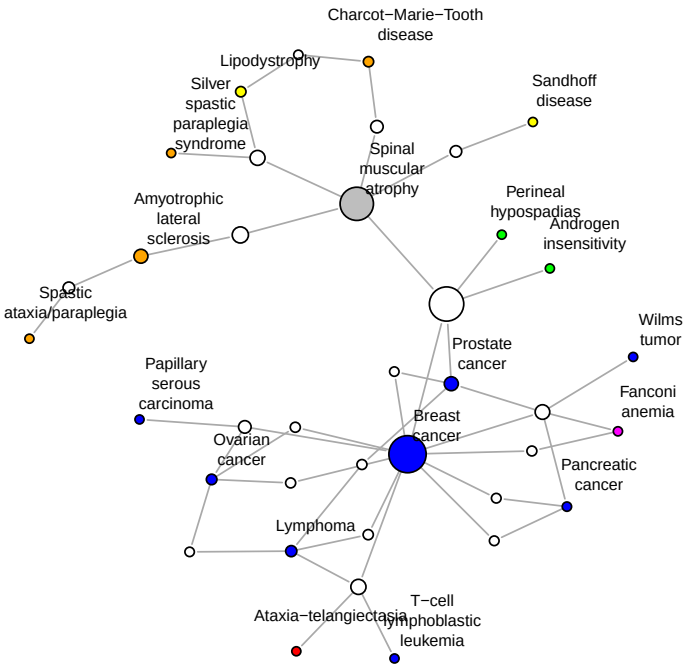

Figure 7: Bipartite gene–disease graph, vertex sizes are scaled by bipartite betweenness centrality.

```
> bipartite.betweenness.centrality(g)

$Bipartite.Betweenness.Centrality
 [1] 0.077113145 0.000000000 0.033191288 0.622408025 0.122349320 0.108816254
 [7] 0.006172840 0.000000000 0.002572016 0.013011648 0.020210287 0.046284021
[13] 0.021557590 0.007870370 0.134564644 0.000000000 0.000000000 0.055555556
[19] 0.108024691 0.000000000 0.157407407 0.536265432 0.024996743 0.558770576
[25] 0.000000000 0.000000000 0.021557590 0.018808455 0.003858025 0.029320988
[31] 0.027006173 0.013011648 0.007844650 0.075617284 0.130401235 0.000000000
[37] 0.055555556 0.000000000
```

As degree centrality is currently not implemented in *igraph*, `degree.centrality` can also be applied to monopartite graphs.

3.3 Centralization

Graph centralization quantifies to which extent a graph resembles a star, that is, contains a highly central vertex around which highly peripheral neighbour vertices are gathered [8]. As discussed in several publications [6, 8], projecting bipartite to monopartite graphs prior to analysis leads probably to biased results due to the loss of information associated with the projection scheme. Therefore, applying single mode centralization measures that determine the extent to which vertices in one subset are central relative only to other vertices within the same subset avoids this problem.

Based on the adapted vertex centrality measures, *biGraph* provides three single mode graph centralization measures.

```
> single.mode.degree.centralization(g)

$Single.Mode.Degree.Centralization.TRUE
[1] 0.4969136

$Single.Mode.Degree.Centralization.FALSE
[1] 0.1450617

> single.mode.closeness.centralization(g)

$Single.Mode.Closeness.Centralization.TRUE
[1] 0.502622

$Single.Mode.Closeness.Centralization.FALSE
[1] 0.3841707

> single.mode.betweenness.centralization(g)

$Single.Mode.Betweenness.Centralization.TRUE
[1] 0.5725384

$Single.Mode.Betweenness.Centralization.FALSE
[1] 0.5053655
```

For the given example, we observe from the results that according to all three centralization measures disease vertices are more centralized than gene vertices.

References

[1] Csardi G, Nepusz T (2006) The igraph software package for complex network research. *InterJournal Complex Systems* p. 1695.

[2] Montañez R, Medina MA, Solé RV, Rodríguez-Caso C (2010) When metabolism meets topology: Reconciling metabolite and reaction networks. *BioEssays : news and reviews in molecular, cellular and developmental biology* 32:246–56.

[3] Vogt I, Mestres J (2010) Drug-Target Networks. *Molecular Informatics* 29:10–14.

[4] Klamt S, Haus UU, Theis F (2009) Hypergraphs and Cellular Networks. *PLoS Comput Biol* 5:e1000385.

[5] Latapy M, Magnien C, Del Vecchio N (2006) Basic Notions for the Analysis of Large Affiliation Networks / Bipartite Graphs. *cond-matstat-mech* .

[6] Vogt I, Mestres J (2010) Assessing information loss associated with bipartite network projection. *submitted* .

[7] Newman M (2001) Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality. *Physical Review E* 64.

[8] Borgatti SP, Everett MG (1997) Network analysis of 2-mode data. *Social Networks* 19:243–269.