DATA 609 Assignment 7: Graph Theory

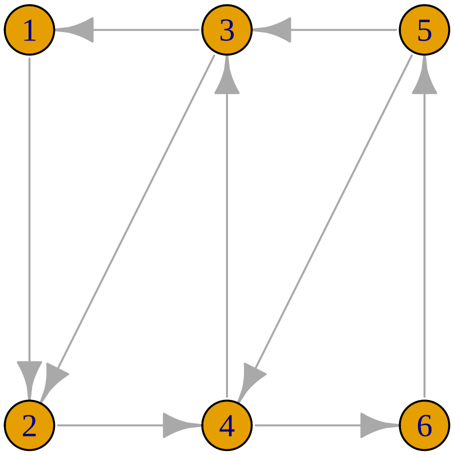
Dan Smilowitz

March 17, 2017

Section 8.1, Problem 2

The graph G is not Eulerian – vertices 2, 4, and 5 have 3 incident edges, which violates the necessary condition of even degree for Eulerian graphs.

With the requirement relaxed, it is possible to travel each bridge exactly once, provided that the starting and ending nodes are both of odd degree. One example, starting at point 5 and ending at point 2, is shown below:



Section 8.2, Problem 1

$$E(G) = \{ab, ae, af, bc, bd, cd, de, df, ef\}$$

Edges ab, bc, and bd are incident with vertex b.

Vertices b and d are adjacent to vertex c.

Three edges (ab, ae, and af) are incident with vertex a, so

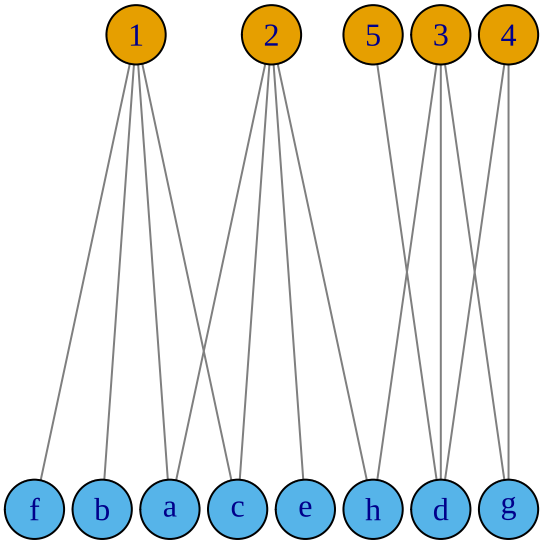
$$\deg(a) = 3$$

There are 9 edges in $E(G)$, as shown above, so

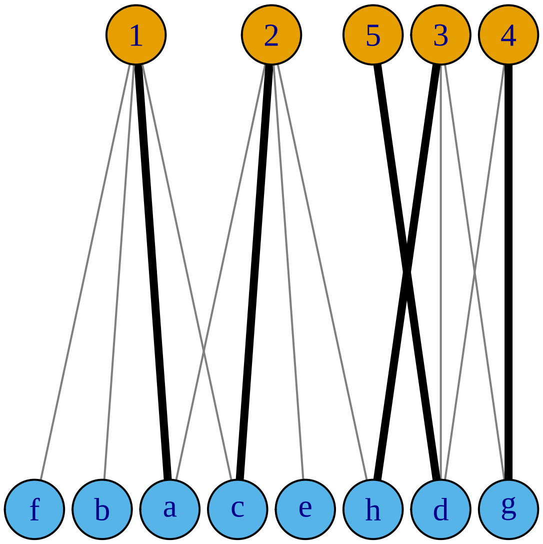
$$|E(G)| = 9$$

Section 8.3, Problem 10

The bipartite graph of the players and positions they can play is presented below:



The only player that can play center (5) is Deb, so she must play that position. With Deb playing center, the only remaining player that can play power forward (4) is Gladys. With Gladys playing power forward, the only remaining player that can play small forward (3) is Hermione. The remaining players can be matched to the two guard positions in 10 different ways – thus there are 10 potential starting lineups, each with Deb at center, Gladys at power forward, and Hermione at small forward. One such lineup is presented below:



If Hermione is unable to play small forward, the coach will not be able to field a feasible starting lineup – in this situation, only Deb or Gladys could play positions 3, 4, or 5, and two players can not fill three positions.

Section 8.4, Problem 1

Visual inspection of the graph suggests that the shortest path is $a - b - d - g - e - i - j$, having a total cost of 12. This can be verified using the `igraph` package:

```
library(igraph)
g <- graph_from_data_frame(data.frame(
  from = c('a', 'b', 'c', 'd', 'a', 'c', 'f', 'b', 'e', 'e', 'e', 'g', 'h', 'i'),
  to = c('b', 'd', 'e', 'g', 'c', 'f', 'i', 'e', 'i', 'g', 'h', 'j', 'j', 'j'),
  weight =  c(2, 2, 4, 2, 4, 2, 6, 7, 3, 1, 2, 8, 4, 2)
), directed = FALSE)

get.shortest.paths(g, 'a', 'j')
```

```
+ 7/10 vertices, named:
[1] a b d g e i j
```

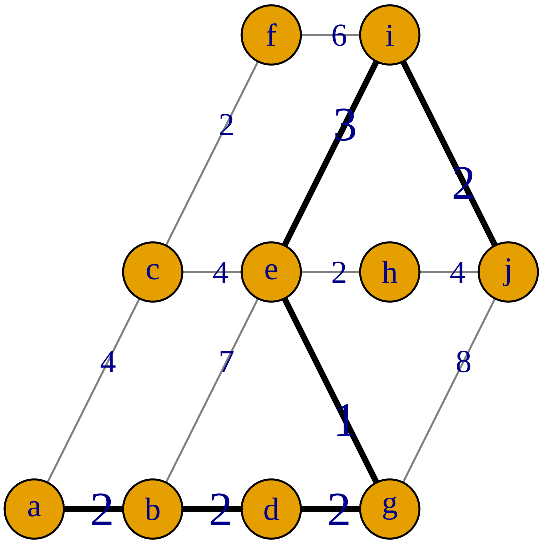
The cost of the shortest path can be calculated with Dijkstra's algorithm. Starting from point a, $L(V) = (0^*, 2, 4, \infty, \infty, \infty, \infty, \infty, \infty, \infty) \rightarrow L(V) = (0^*, 2^*, 4, \infty, \infty, \infty, \infty, \infty, \infty, \infty)$

Proceeding with the algorithm, the final result is

$$L(V) = (0^*, 2^*, 4^*, 4^*, 6^*, 7^*, 6^*, 9^*, 10^*, 12^*)$$

This confirms that the cost of the shortest path from a to j has a cost of 12.

The shortest path is presented graphically below.



Section 8.4, Problem 3

Beginning with the path $s - x_1 - y_2 - t$, flow f_c is 1, and the edges sx_1 and y_2t now have capacity 0. Next, the path $s - x_2 - y_6 - t$ is taken. The edges sx_2 and y_6t now have zero capacity, and $f_c = 2$. Next, the path $s - x_3 - y_1 - t$ is taken – edges sx_3 and y_1t have zero capacity and $f_c = 3$. The next path taken is $s - x_4 - y_3 - t$ – edges sx_4 and y_3t have zero capacity and $f_c = 4$. There are no remaining directed paths from s to t with the remaining capacity, thus **the maximum flow is 4** – this coincides with the maximum matching for the bipartite graph.

Section 8.5, Problem 4

$$\text{Maximize } z = x_{sa} + x_{sb}$$

Subject to

$$\begin{gathered} x_{sa} \geq 0 \\ x_{sb} \geq 0 \\ x_{ab} \geq 0 \\ x_{ac} \geq 0 \\ x_{bc} \geq 0 \\ x_{bd} \geq 0 \\ x_{cd} \geq 0 \\ x_{ct} \geq 0 \\ x_{dt} \geq 0 \end{gathered}$$

$$\begin{gathered} x_{sa} \leq 3 \\ x_{sb} \leq 5 \\ x_{ab} \leq 2 \\ x_{ac} \leq 6 \\ x_{bc} \leq 2 \\ x_{bd} \leq 4 \\ x_{cd} \leq 1 \\ x_{ct} \leq 4 \\ x_{dt} \leq 5 \end{gathered}$$

$$\begin{gathered} x_{sa} = x_{ab} + x_{ac} \\ x_{sb} + x_{ab} = x_{bc} + x_{bd} \\ x_{ac} + x_{bc} = x_{cd} + x_{ct} \\ x_{bd} + x_{cd} = x_{dt} \end{gathered}$$