Notes on Centrality

Shambhavi Suryanarayanan, Elizaveta Rebrova

Spring 2024

Centrality measures provide a way to quantify the how important a node or an edge is in a graph. There are different ways of quantifying this "importance" giving rise to different notions of "centrality". The goal of this note is to review the different centralities used in the study of networks.

1 Connections-based centrality measures

1.1 Degree and eigenvector centralities

Definition 1.1 (Degree Centrality)**.** The *degree centrality* of a vertex v in a undirected graph $G = (V, E)$ is given by the degree of that vertex.

This is a simple measure of how "important" a vertex is based on the number of connections (adjacent vertices, neighbours) it has. However, this is a local measure. Here, all neighbours are considered to be equal. We can further refine this idea by taking into account the importance of the neighbours too. One such way this is done is by using the eigenvector centrality.

Definition 1.2 (Eigenvector Centrality)**.** Consider an undirected connected graph $G = (V, E)$ with an adjacency matrix given by A. Let it's leading eigenvalue be λ_{max} and the leading eigenvalue be $\mathbf{x}$. That is,

$$A\mathbf{x} = \lambda_{max}\mathbf{x}.$$

Then, the eigenvector centrality of a vertex v is given by $\mathbf{x}(v)$. The Perron-Frobenius theorem (stated in class) guarantees that this value is unique and positive.

Nodes with high eigenvector centrality are ones which have a vary large degree and/or are connected to important vertices.

Extending this notion to directed graphs has to be performed carefully. Here,as the adjacency matrix may no longer be symmetric, we first need to fix whether the in-degree or out-degree is of interest. Motivated by real life networks such as the World Wide Web and the citation networks, it is clear that we should be looking at the in-degree in such graphs. So we are interested in the left eigenvectors of A, or equivalently, the right eigenvectors of $A^\intercal$.

However, we are faced with another roadblock as the Perron-Frobenius theorem will not hold unless $A^\intercal$ is strongly connected. This means, that for some class of directed graphs, the eigenvector centrality of a node can be zero despite having a positive in-degree. In fact, the eigenvector centrality of any node in a directed acyclic graph is 0. This problem is addressed by the following defintions of eigenvector based centralities.

1.2 Katz Centrality

Definition 1.3 (Katz Centrality)**.** Consider a directed graph $G = (V, E)$ with adjacency matrix A. Further, let α and $\beta > 0$ be two parameters. Then the Katz centrality of nodes is given by the vector x which is the solution the following equation -

$$x_i = \sum_j A_{ji}x_j + \beta \quad \forall i \iff x = \alpha A^\intercal x + \beta \cdot \mathbb{1}$$

When α is chosen such that $(I - \alpha A^\intercal)$ is invertible, then

$$x = \beta(I - \alpha A^\intercal)^{-1}\mathbb{1}$$

One drawback faced in using Katz centrality is that if an edge with high Katz centrality points to many edges, all of them inherit a centrality values. However, in many contexts, it would help to devalue the centrality of a vertex if it is one of the many ones pointed to by a high Katz-centrality vertex. This is what PageRank, the algorithm used by google to sort search results hopes to achieve. This is done weighting the adjacency matrix's entries with the out degree appropriately.

1.3 PageRank Centrality

Definition 1.4 (PageRank Centrality)**.** Consider a directed graph $G = (V, E)$ with adjacency matrix A. Further, let α and $\beta > 0$ be two parameters. Then the PageRank centrality of nodes is given by the vector x which is the solution the following equation -

$$x_i = \sum_j \frac{A_{ji}}{\text{out-deg(j)}} x_j + \beta \quad \forall i \iff x = \alpha A^\intercal D^{-1} x + \beta \cdot \mathbb{1}$$

where D is the augmented out-degree matrix. That is, it is a diagonal matrix with the diagonals coinciding with the out-degree for vertices with outgoing edges . And for vertices v with no edges going out, $d_{vv} = 1$. When α is chosen such that $(I - \alpha A^\intercal D^{-1})$ is invertible, then

$$x = \beta(I - \alpha A^\intercal D^{-1})^{-1}\mathbb{1}$$

Remark 1. Here, setting $d_v v = 1$ when out-degree(v) is 0 doesn't affect our analysis of the network as when this is the case $A_{vi} = 0$ for all $i \in V$.

1.4 Hubs and Authorities (HITS centrality)

In directed networks, a vertex can be seen as performing two kinds of duties -

- One, as an *authority*, a node which is pointed to by nodes with high centrality
- And, as a *hub*, a node that points you towards the "important" nodes, the authorities

If we were to look at the citation network, the seminal papers would be the authorities while important surveys which point to it would be hubs. A good hub points to more authorities, and pointing to an authority should increase the hub-score. From this it is clear that both these score influence each other.

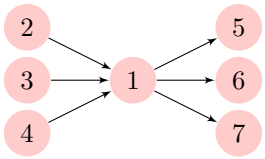


Figure 1: Toy network

Before defining, the hub and authority centralities, let's look at a toy case as in Figure 1.4. Let the authority scores be given by x and hub scores by y. Let α and β be positive constants. Let's say we know the hub-scores y. Then the authority score of vertex 1 is given by α times the sum of the hub score of edges pointing to it

$$x_1 = \alpha(y_2 + y_3 + y_4) \implies x_1 = \alpha \sum_{j \in V} A_{ji} y_j.$$

Here, let's say we know the authority scores x. Then the hub score of vertex 1 is given by β times the sum of the authority score of edges it points to

$$y_1 = \beta(x_5 + x_6 + x_7) \implies y_1 = \alpha \sum_{j \in v} A_{ij} x_j.$$

Generalising to all vertices, we get that the hubs and authority score vectors are given by solutions to the equations -

$$x = \alpha A^\intercal y, \tag{1}$$
$$y = \beta A x \tag{2}$$

Plugging one into the other, we get that x and y should satisfy -

$$x = (\alpha\beta)\ A^T A\ x \quad \& \quad y = (\alpha\beta)\ AA^\intercal\ y.$$

As noted in class, $A^\intercal A$ and $AA^\intercal$ are both symmetric and share the same spectrum. So we can choose $\alpha\beta$ such that $(\alpha\beta)^{-1}$ is the largest eigenvalue of one of the matrices. Then, the hubs and authorities centralities can be more formally defined in terms of the leading eigenvectors as follows -

Definition 1.5 (Hubs and Authorities Centrality a.k.a. HITS)**.** Consider a directed graph $G = (V, E)$ with an adjacency matrix A. The authority centrality x and the hub centrality y is given by the leading/principal eigenvectors to the matrices $A^\intercal A$ and $AA^\intercal$ respectively. That is,

$$A^\intercal A x = \lambda_1 x, \tag{3}$$
$$AA^\intercal y = \lambda_1 y. \tag{4}$$

where λ_1 is the leading eigenvalue of $A^T A$ and $AA^\intercal$

Remark 2. Both HITS scores and PageRank values can be computed using power iteration methods and will be discussed later in class

2 Location-based centrality measures

2.1 Closeness and harmonic centralities

Another way to quantify an importance of a node is to based on how close other nodes are to it. This is metric of *closeness* is important in many real-world networks like road networks and social media networks. In the former, a vertex which is close to many other nodes will prove pivotal for transit. Similarly, in a social network, such nodes with high closeness determine how information spreads through the network. Formally, we can define the *closeness centrality* of a node as inverse of the average distance to other vertices.

Definition 2.1 (Closeness Centrality)**.** Consider a connected graph $G = (V, E)$ on N vertices. For any pair of vertices $u, v \in V$, let $dist(u, v)$ denote the shortest path between u and v. The *closeness centrality*, $c(v)$ of any node $v \in V$ is given by

$$c(v) := \frac{N-1}{\sum_{u \in V \setminus \{v\}} dist(u, v)}$$

Example. Let's look at how the closeness centrality of the node b. In the graph below can be computed. The shortest path to other vertices are recorded in the table below.

v	$dist(g, v)$
d	1
c	1
a	1
e	2
f	3
g	3

This gives us that closeness centrality of b here is $\frac{1}{\text{average distance to other vertices}} = \frac{6}{11}$

When working with graphs which aren't connected, the distance between two vertices u, v in different connected components would be $dist(u, v) = \infty$. To accommodate such cases, another similar notion of centrality called the *harmonic centrality* can be defined. This is given by -

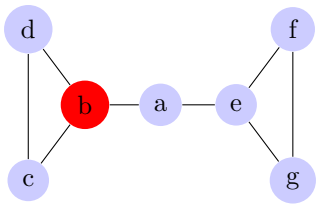


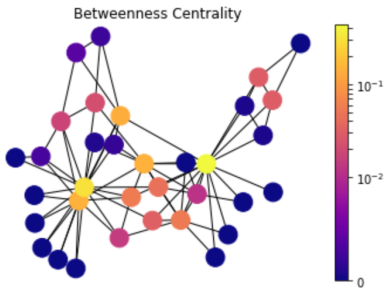
Figure 2: Closeness centrality of node b

Definition 2.2 (Harmonic Centrality)**.** Consider a graph $G = (V, E)$ on N vertices. The *harmonic centrality*, $c(v)$ of any node $v \in V$ is given by

$$c(v) := \frac{1}{N-1} \sum_{u \in V \setminus \{v\}} \frac{1}{dist(u, v)}$$

2.2 Betweenness Centrality

Betweenness centrality is a widely used measure that captures *an importance of a vertex/edge in a network in allowing information to pass from one part of the network to the other*. It is defined based on the amount of shortest paths within the network that has to pass through the given vertex. Informally, network "bottlenecks" (including bridges and local bridges) have higher betweenness values. The picture below is due to Can Güney Aksakalli.



Definition 2.3 (Betweenness Centrality for nodes)**.** Consider a graph $G = (V, E)$. For $s, t, v \in V$ let $\sigma_{s,t}$ denote the number of shortest paths from s to t and $\sigma_{s,t}(v)$ denote the number of shortest paths from s to t that pass through v. Then, the betweenness of node v, denoted by $h(v)$ is given by:

$$h(v) := \frac{1}{2} \sum_{s,t \in V s,t \neq v} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}.$$

- For the betweenness of a node v, we do not consider paths that originate or end at v.
- The factor of $\frac{1}{2}$ appears because we if we are considering a path p from s to t, we would be accounting for it twice, once in $\sigma_{s,t}$ and again in $\sigma_{t,s}$. Instead, one can as well only consider unordered pairs (s, t) from V. *Overall, in this definition, each pair of vertices s and t (so that both s and t are distinct from v) should be considered once.*
- The definition above is consistent with the examples we calculated in class, see also the example below.

  However, in literature, it is typical to normalize the definition of betweenness in other ways accounting for the total size of the network (note that the values of $h(v)$ tend to be larger not only in reflection of the comparatively important role of a vertex, but also in reflection of the total network size). Two alternative popular ways to rescale the betweenness value are

1. to define $h_1(v)$ by dividing $h(v)$ through by the number of pairs of nodes inn the network not including v (so, by $(n-1)(n-2)/2$ for undirected networks wih n vertices). This way, $h_1(v) \in [0,1]$ for all v. But the values of $h_1(v)$ can be very small in large sparse netwoks (and real-life networks tend to be very sparse!)
2. to normalize based on the maximal and minimal values of $h(v)$, that is

$$h_2(v) := \frac{h(v) - \min_v h(v)}{\max_v h(v) - \min_v h(v)}$$

Check that in this case we also ensure that $h_2(v) \in [0,1]$ for any v and we do not shrink the range of values of the betweenness function as much as in the defnition of $h_1(v)$.

Finally, note that rescalings do not matter if you only need to compare the betweennesses of different vertices within one network. For the exams, you only need to know the Definition 1.1 of $h(v)$, not other possible ways to rescale it.

- The function for computing betweenness in the NetworkX package in python uses a slightly different definition of betweenness than what we consider. Pre-defined functions for computing centrality measures cannot be used in the exams.

We can define betweenness centralities for edges of a graph in a similar fashion.

Definition 2.4 (Betweeness centralities for edges)**.** The betweenness centrality measure for the edge $e = (u, v)$ can be defined as

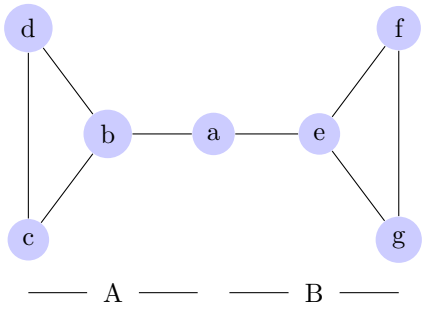
$$h_e(u - v) := \frac{1}{2} \sum_{s,t \in V} \frac{\sigma_{s,t}(e)}{\sigma_{s,t}}$$

with the same agreement on normalization (each shortest path in the graph is computed once).

Remark 3. Note, that the endpoints of an edge e are included in the summation. If we consider an edge $e = (u, v)$, then $\frac{\sigma_{u,v}(e)}{\sigma_{u,v}} = \frac{\sigma_{v,u}(e)}{\sigma_{v,u}} = 1$. This implies that the edge betweenness is strictly positive (in fact, it is at least 1) for every edge while the node betweenness can be zero for some nodes.

3 Example: Calculating Betweenness centralities

Calculating node betweeness:



Let's consider the betweenness of the node a. Let the nodes on the left of a, be denoted by A and those on the right by B. For any pairs of nodes s, t in A, none of the shortest paths would pass through a. Hence, $\sigma_{s,t}(a) = 0$. This similarly holds for pairs of node in B.

Now consider a node $s \in A$ and $t \in B$. All of the shortest paths from s, t would have to pass through v. As removing a would make A and B disconnected. We can also note that here, all the shortest paths will be unique. Thus $\sigma_{s,t}(a) = \sigma_{s,t}$. Similar argument holds for $s \in B$ and $t \in A$.

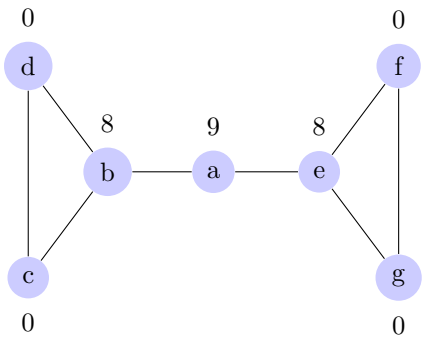


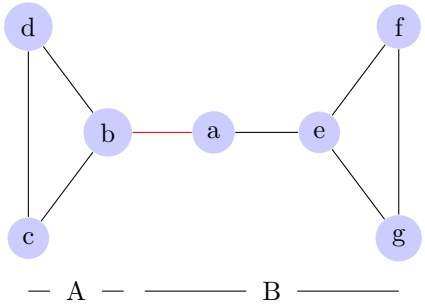
Figure 3: Node betweenness values for all vertices in the graph

Hence,

$$h(a) = \frac{1}{2}\Big(\sum_{s\in A, t\in B} \frac{\sigma_{s,t}(a)}{\sigma_{s,t}} + \sum_{s\in B, t\in A} \frac{\sigma_{s,t}(a)}{\sigma_{s,t}}\Big) = \frac{1}{2}\Big(\sum_{s\in B, t\in A} 1 + \sum_{s\in A, t\in B} 1\Big) = 3\times 3 = 9$$

Similarly, we can compute node betwennesses for all vertices of the graph:

Calculating Edge Betweeness: Now, let's look at how we can compute edge betweenness values for the same graph. Let's first focus on the a-b edge, which we shall denote by e! Similar to what was done in the vertex case, we can partition to the vertices into two classes A and B, based on whether they are to the left or right of the edge respectively. Recall the formula for calculating edge betweeness:



$$\begin{aligned} h_e(e = a-b) = \frac{1}{2}\sum_{s,t\in V} \frac{\sigma_{s,t}(e)}{\sigma_{s,t}} &= \frac{1}{2}\Big(\sum_{s\in A, t\in B} \frac{\sigma_{s,t}(e)}{\sigma_{s,t}} + \sum_{s\in B, t\in A} \frac{\sigma_{s,t}(e)}{\sigma_{s,t}}\Big) \\ &= \sum_{s\in A, t\in B} \frac{\sigma_{s,t}(e)}{\sigma_{s,t}} \\ &= \sum_{s\in\{b,c,d\}, t\in\{a,e,f,g\}} \frac{\sigma_{s,t}(e)}{\sigma_{s,t}} \\ &\underset{(1)}{=} \sum_{s\in A, t\in B} 1 = \ 3\cdot 4 = 12 \end{aligned}$$

Here, Eqn (1) follows as the shortest path from any vertex in A to any vertex in B includes the edge $a-b$. That is, $\frac{\sigma_{s,t}(e)}{\sigma_{s,t}} = 1$ for all $s \in A, t \in B$.

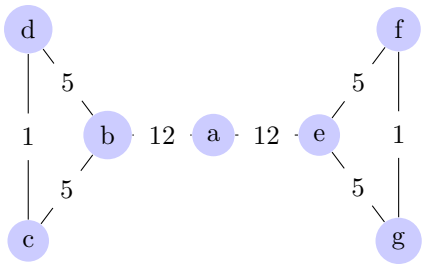


Figure 4: Edge betweenness values for all edges in the graph

In a similar fashion, we can compute the edge betweenness values for all the edges in this graph. These are given by -

Note that this is only a minimal example to illustrate the normalization factor that we use. For less trivial examples see lecture notes and the textbook.

3.1 Calculating betweenness centrality for all edges at once

For small networks, it is enough to use Definition 1.1 to compute betweenness centrality. If you deal with a large network, one needs to find *all* shortest paths between *all* vertices, and this can be too slow. A modification of the BFS (breadth first search) algorithm can address this issue. A version of this algorithm that is used for computing *edge betweenness* is presented here (also in the textbook in Section 3.6).

This algorithm is based on an equivalent representation of this problem is through the idea of flows. In this formulation, you assume that there is one unit of flow between each par of vertices in the graph. This one unit of flow is equally divided amongst all it's shortest paths. So, the total flow on each edge would correspond to the edge betweenness of the corresponding edge.

The following algorithm is described on the pages 80-81 of the Easley&Kleinberg textbook:

1. Starting from every vertex,
   (a) Construct the BFS (breadth first search) tree,
   (b) In the tree, compute *potentials of all vertices* as numbers of (shortest) paths going strictly down from the root to a given vertex. This is worked out top to bottom, for example, the vertices in the first layer of the BFS tree have potentials 1.
   (c) Then, compute *edge flows* as the number of units needed to bring 1 unit of flow to each vertex on the bottom layer of the BFS tree while leaving 1 unit of flow at every vertex along the downward paths. This is worked out bottom to top, the flow splits between the incoming (from the top) edges proportionally to the previous vertex potentials. For the horizontal edges, the flow is zero.
2. Add up flows found on Step 1 for every edge (over all BFS trees) to almost get betweenness measure for each edge.
3. Divide all the numbers by 2 to remove double-counting paths from s to t and from t to s.

We note that the algorithm searches for the betwennesses of all edges at once, which aligns with the needs of Girvan-Newman algorithm for betweenness-based clustering. However, in order to find the edges with maximal betweenness (as for Girvan-Newman algorithm), the normalization does not matter and, in particular, step 3 can be skipped.

Next, we show the same example as above computed via the BFS approach. Starting from a, we can perform BFS and use this to count paths or *potential* by going down the layers. This is denoted in the color

green in Fig. 3.1 below. Now that we have the number of paths, starting at the bottom most layer, we work our way up to compute edge flows (denoted in red). If we were to start with the $a - d$ path, the edge flow for $b - d$ would be one. Similarly, for $b - c$, $e - f$ and $e - g$ edge.

Now, moving on to the second layer, if we were to look at $a - b$ edge, it would have a flow of 3, where one is from the path $a - b$ itself, and 2 from the sum of edge flows from the bottom layers. We can repeat this to compute the edge flows, for all edges in this BFS representation. Here, the edge flow value of an edge e in the BFS from a is nothing but the following quantity

$$\sum_{t \in V \setminus \{a\}} \frac{\sigma_{at}(e)}{\sigma_{at}}$$

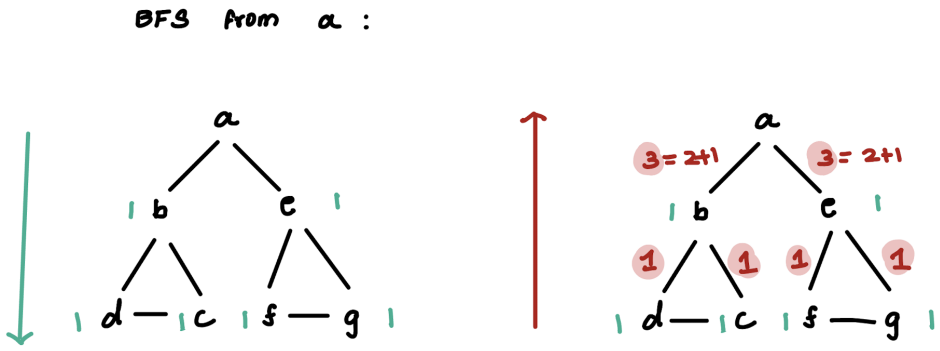


Figure 5: Edge flow values corresponding to BFS done starting from node a

To compute the betweenness centrality, we would need -

$$h_e(e) = \frac{1}{2} \sum_{a \in V} \sum_{t \in V \setminus \{a\}} \frac{\sigma_{at}(e)}{\sigma_{at}}$$

To accomplish this, write down the BFS trees from all the vertices and compute the corresponding edge flows as depicted in Figures 3.1and 3.1. Then, the edge flow values are summed up and divided by 2 to get the betweenness centrality of each edge. For this example, this would be as given below. Note that they match with our previous computations.

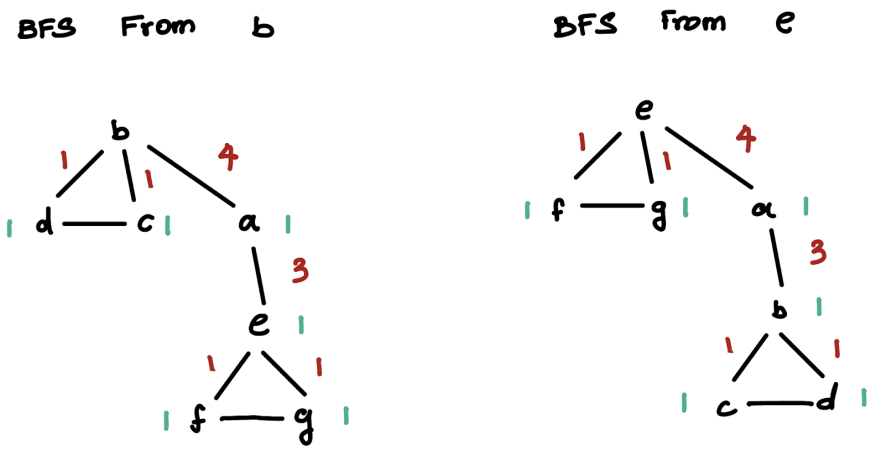


Figure 6: Edge flow values corresponding to BFS done starting from node b and e

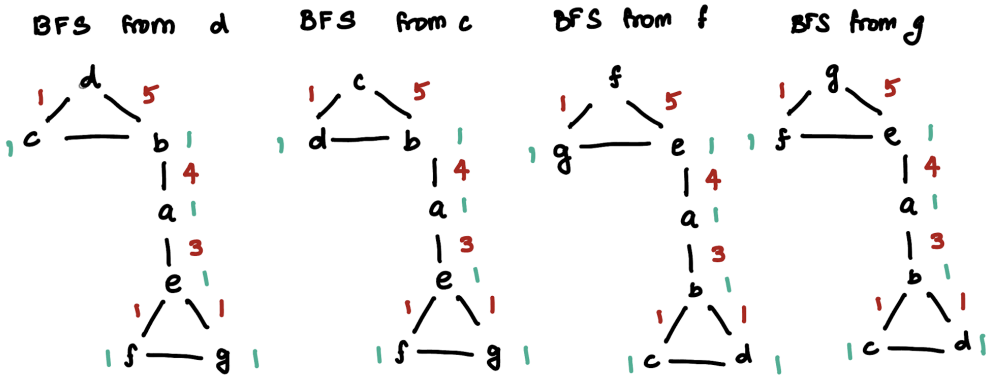


Figure 7: Edge flow values corresponding to BFS done starting from node c,d,f and g

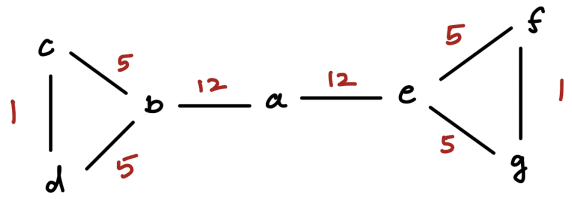


Figure 8: Betweenness centrality values obtained by summing up edge flows and dividing by 2