1 a) My Mapreduce procedure :

i) First step: Mapping all the inbound edges/targets with its weight in the form of tuple. This is achieved by Map function. This function delivers intermediary records of key-value pairs.

ii) Second step: Reduces the intermediary records of key-value pairs that share same key into a smaller set. This is achieved by Reducer function.

Example:

src	tgt	weight
117	51	1
194	51	1
299	51	3
230	151	51
194	151	79
51	130	10

Consider the above graph data,

When the above graph data is passed to the Map function it generates key value pairs as shown below:

(51,1)
(51,1)
(51,3)
(151,51)
(151,79)
(130,10)

The map function reads each and every line of the graph data, stores the target node value and the weight of the node in intermediary records

Now when the generated intermediary records shown above are passed to the reducer function, the function retains the key-value pair that has maximum weight value. For all the records sharing the same key, the reducer function retains only that record that has maximum weight.

So the output of reduce function will be
(51,3)
(151,79)
(130,10)

1 b) MapReduce (neighbours of neighbours) algorithm:

To find the vertices that are two hops away from the vertex, I am using a Mapreduce algorithm and the procedure is :

i) Step 1: Every source is mapped with it's target and every target is mapped with -1*source. The -1 multiplication to the source node ID is just to indicate outward direction of edge. All the (source,target) and (target,-source) are stored in temporary record of key-value pair.

ii) Step 2: The reducer groups all the values of the mapped key-value pairs with same key as group of two when the values are of same sign.

src	tgt
4	3
1	2
2	3
4	2
2	1
3	2

For the given example above, the map function produces the below output of (source,target) and (target,-source) as
(4,3) (1,2) (2,3) (4,2) (2,1) (3,2) (3,-4) (2,-1) (3,-2) (2,-4) (1,-2) (2,-3).

The reducer then generates pairs of value-value pairs when keys match and the sign of both values is same. Ie., when reducer finds (4,3) and (4,2), it generates (3,2) and (2,3) and they are connected through 4. Similarly when reducer finds (2,-1) and (2,-4) it creates output pairs of (1,4) and (4,1) and 4 and 1 are connected through 2.

The output of the reducer is as shown below: