Marden Clustering Algorithm

```
function ApplyAlg(network G_orig)
    allClusters ← []
    G ← G_orig
    change ← true
    while change do
        L ← list of G's nodes in order of highest to lowest node degree
        C ← []
        while L not empty do
            mark all nodes of G as unchecked
            seed ← first element of L
            seedAdjList ← list of nodes adjacent to seed
            cluster ← [seed]
            cluster ← FindClusters(seed, seed, seedAdjList, cluster)
            remove all nodes in cluster from L
            add cluster to C
        end while
        G_new ← condensed network of G where the nodes are the clusters in C
        if G == G_new then
            change ← false
            append [C] to allClusters
        else
            G ← G_new
        end if
    end while
    return allClusters
end function

function FindClusters(n, seed, seedAdjList, cluster)
    for each node nAdj adjacent to n do
        if nAdj is unchecked then
            mark nAdj as checked
            score ← FindScore(nAdj, seed, seedAdjList)
            if score ≥ 0 then
                add nAdj to cluster
                FindClusters(nAdj, seedAdjList, cluster)
            end if
        end if
    end for
    return cluster
end function

function FindScore(n, seed, seedAdjList)
    score ← 0
    seedAdj ← false
    for each node nAdj adjacent to n do
        if nAdj in seedAdjList then
            score ← score+weight of edge connecting nAdj and n
        else
            score ← score−weight of edge connecting nAdj and n
        end if
        if nAdj == seed then
            seedAdj ← true
        end if
    end for
    if seedAdj and seed not in seedAdjList then
        score ← score+weight of edge connecting n and seed
    end if
    return score
end function
```