Algorithm 1 EDGE_DEL_MAIN(G)

1: cost = cost ← TRANSITIVE_CLOSURE_COST(G);
2: if (cost == 0) then
3:   return(null, 0);
4: end if
5: actions ← null; delcost ← 0;
6: while (G is still connected) do
7:   uv ← REMOVE_CULPRIT(G);
8:   actions.add(uv);
9:   delcost+ = s(uv);
10: end while
11: // Adjust actions such that it only contains the edge removals contributing to the separation of two subgraphs of G
12: //Assume G is cut into G_1 and G_2
13: while (uv in actions) do
14:   if (both u, v are in G_1 or G_2) then
15:     actions.remove(uv);
16: end if
17: end while
18: //Solve the problem in a recursive manner for G_1 and G_2, until no better solution can be found
19: if (delcost >= cost) then
20:   return(null, cost);
21: end if
22: (list1, cost1) ← EDGE_DEL_HEURISTICS(G_1);
23: if (delcost + cost1 >= cost) then
24:   return(null, cost);
25: (list2, cost2) ← EDGE_DEL_HEURISTICS(G_2);
26: end if
27: if (delcost + cost1 + cost2 >= cost) then
28:   return(null, cost);
29: end if
30: actions.add(list1);
31: actions.add(list2);
32: //add all the edge insertions required for the closure of transitivity
33: actions.add(insertions);
34: return(actions, delcost + cost1 + cost2);

Additional Table 1 - The details of the 86 putative GWAS associations