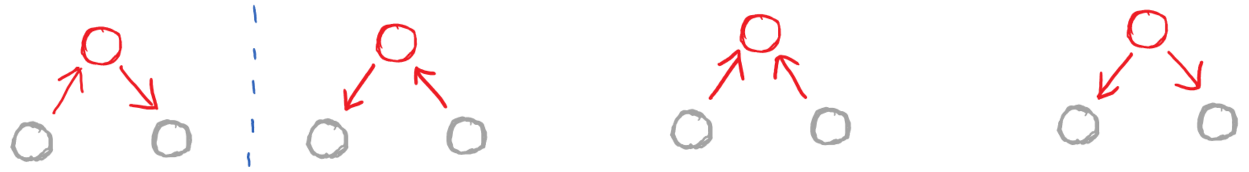
Grogan Huff 2016/4/06

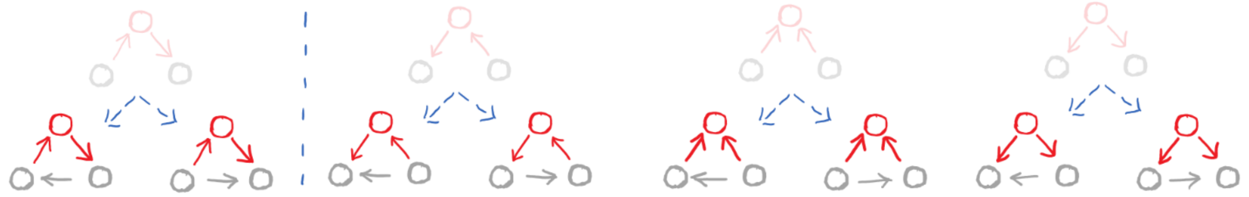
The maximum amount of directionless relationships possible among $n$ nodes in graph is given by the equation $\frac{n*(n-1)}{2}$. For a fully connected triangular relationship among 3 nodes there are 3 relationships, one between each pair of nodes. The graph can't be more interconnected, so it has a clustering coefficient of 100%. Relationships may exist from nodes of the triangle to nodes not participating in the triangle, but we can safely ignore them when focusing on the triangle.

The graph database platform we are using, Neo4j, stores relationships with direction, but in reality gene co-expression relationships are directionless. If gene `(A)` is `-[co-expressed]->` with `(B)`, then the relationship in the reverse direction is also true. However, by having arbitrary directionality in the storage of our data and taking care to prevent the formation of redundant directional relationships between two nodes, it becomes possible to construct a search pattern that will find 100% of triangles while also avoiding repeated results like `(A)<-->(B)` and `(B)<-->(A)` from being returned in our queries of the database.

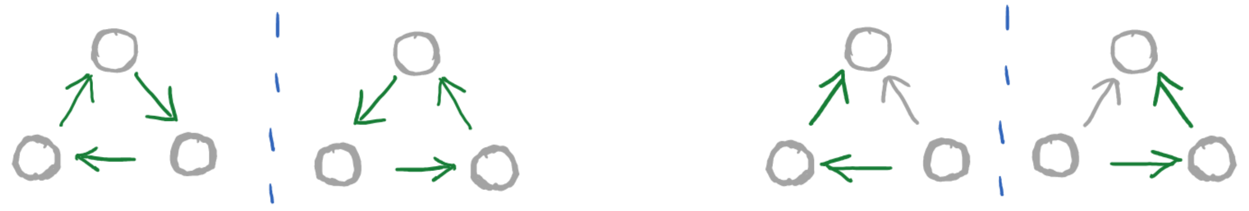
Consider the following:
Given three nodes in a triangular relationship, stored with direction, we can choose one among the three nodes and designate it node O. From node O's perspective, there exists a finite number of possible states for its 2 relationships to the 2 other nodes. Either one relationship is incoming and one relationship is outgoing (this possibility has a mirrored twin), both relationships are incoming, or both relationships are outgoing.



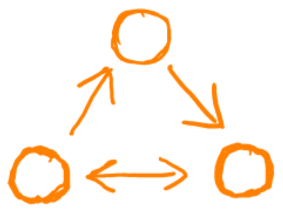
From there, we consider the state of the relationship between the other two nodes which must point in either the left or the right direction (else, we do not have a triangular relationship).



These represent every possible triangle from node O's perspective. While these results may seem daunting, close review reveals only two unique structures, and their mirrors. Either we have a "perfect" triangle where each node has one incoming and one outgoing relationships, or we have a structure with one "bully" node having two incoming relationships, one "selfless" node with two outgoing relationships, and one "balanced" node which looks like any node from a perfect triangle.



The viability of a searching pattern hinges on that "balanced" node. A perfect triangle consists of three balanced nodes and there is no way to differentiate between starting position. Example, a perfect A->B->C-> triangle should return A B C, B C A, and C A B. With non-perfect triangles, the balanced node provides directionality. Because we built our database on a list of sorted nodes, all possible outgoing relationships are created before we move on to the next node. This construction method inhibits the formation of perfect triangles and allows us to search with the following pattern.



Since the third, "wobble" relationship cannot exist in both directions at once and the balance node of the search pattern will always find the balanced node of the triangle, this search pattern returns non-redundant results while also returning 100% of existing triangles.