Journal of Statistical Software

July 2016, Volume 71, Issue 10. doi: 10.18637/jss.v071.i10

Computation of Graphlet Orbits for Nodes and Edges in Sparse Graphs

Tomaž Hočevar
University of Ljubljana

Janez Demšar
University of Ljubljana

Abstract

Graphlet analysis is a useful tool for describing local network topology around individual nodes or edges. A node or an edge can be described by a vector containing the counts of different kinds of graphlets (small induced subgraphs) in which it appears, or the "roles" (orbits) it has within these graphlets. We implemented an R package with functions for fast computation of such counts on sparse graphs. Instead of enumerating all induced graphlets, our algorithm is based on the derived relations between the counts, which decreases the time complexity by an order of magnitude in comparison with past approaches.

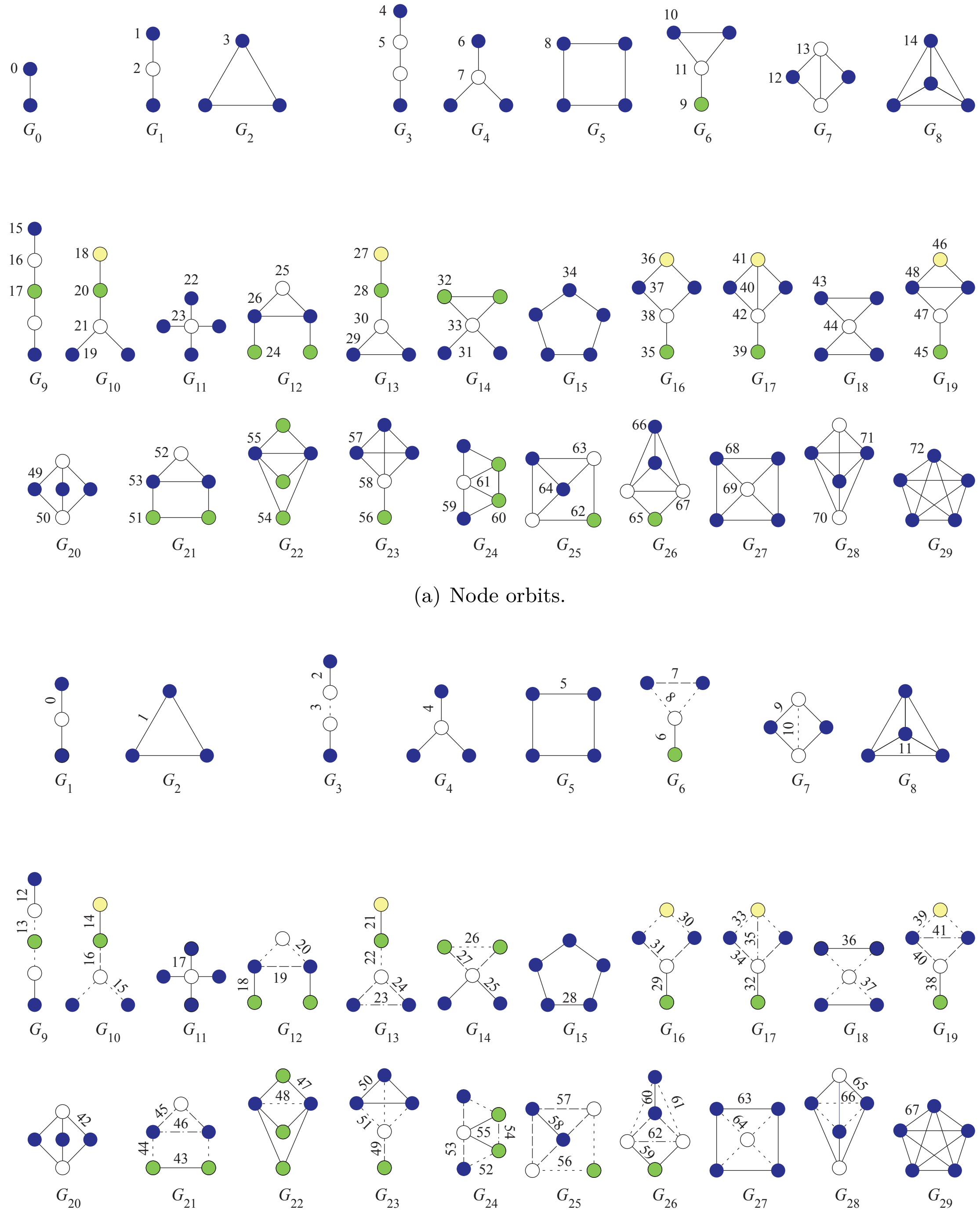
Keywords: network analysis, graphlets, data mining, bioinformatics.

1. Introduction

Analysis of networks plays a prominent role in many areas of science and business, from genetic and protein networks in bioinformatics to social networks in mining user data. Describing the roles of individual nodes and edges, clustering them, and predicting their future development requires observing their locally defined properties. One of the methods – used particularly in bioinformatics – is based on counting graphlets and graphlet orbits.

Graphlets are small connected simple graphs (Pržulj, Corneil, and Jurisica 2004). There are 9 different graphlets with two to four nodes and 30 graphlets with up to five nodes. In graphlet-based network analysis, we examine induced graphlets within the network: For each node, we count the number of times the node touches an induced graphlet of each kind, which gives a 9- or 30-dimensional vector description of the local topology surrounding the observed node. One of the vector components represents, for instance, the number of times the node is included in an induced star on five nodes.

Furthermore, we can group nodes of each graphlet into orbits (Pržulj 2007) with respect to



(a) Node orbits.

(b) Edge orbits.

Figure 1: Graphlets with 2–5 nodes with enumeration of orbits. Node colors and line shapes, which are chosen arbitrarily, correspond to orbits within each graphlet. Node orbits are enumerated as in Pržulj (2007); edge orbit numbers are enumerated by increasing orbits of the corresponding node pairs.

the graphlet automorphisms (Figure 1(a)). Orbits define the "roles" of the nodes within the graphlet. For instance, in a star on five nodes (G_{11}), one node represents the center and the remaining four nodes are the leaves; the nodes of the star thus form two different orbits (numbered 23 and 22, respectively). Instead of counting only the number of appearances of induced stars that touch an observed node in the network, we can count how many times the node represents the center of such star (i.e., the node is connected to four nodes that are not connected to each other) and how many times it has the role of a leaf (i.e., it is connected to a node that is connected to another three nodes that are disconnected from each other and from the observed node). This gives a finer description of the node's vicinity with a 15-dimensional vector for four-node graphlets and a 73-dimensional vector for five node graphlets.

Similar can be done for edges (Solava, Michaels, and Milenković 2012): There are 68 edge orbits for graphlets with 3–5 nodes, which allow for a characterization of an edge with a 68-dimensional vector (Figure 1(b)).

Figure 2 gives an illustration for a small network. Figure 2(a) shows the network, and Figures 2(b), 2(c) and 2(d) show all four-node subgraphs that include node C; node C appears in orbits 5, 7 and 11. Table 2(e) shows all orbit counts for node C, including those belonging to two- and tree-node subgraphs. Table 2(f) shows the orbit counts for all nodes and orbits. The vector (row) corresponding to node C is quite different from others (e.g., counts for orbits 2, 7, 11), which indicates its special place in the graph. Signatures of A and B, on the other hand, are the same since the two nodes map to each other in an automorphism of the graph.[1]

The straightforward computation of orbit counts by enumeration takes $O(nd^{k-1})$ time, where n is the number of nodes (typically thousands or tens of thousands), d is the maximal node degree (usually up to one hundred), and k is the graphlet size (4 or 5). We have recently presented a combinatorial approach for counting orbits of nodes (Hočevar and Demšar 2013) in time that is, for practical purposes, proportional to nd^{k-2}. Using this technique, the common-size networks from proteomics can be analyzed in a reasonable time of a few hours on a common desktop computer. In this paper, we provide the first complete description of the algorithm, including its novel extension to counting edge orbits (Section 2), and then document the corresponding R package together with two usage examples (Section 3).

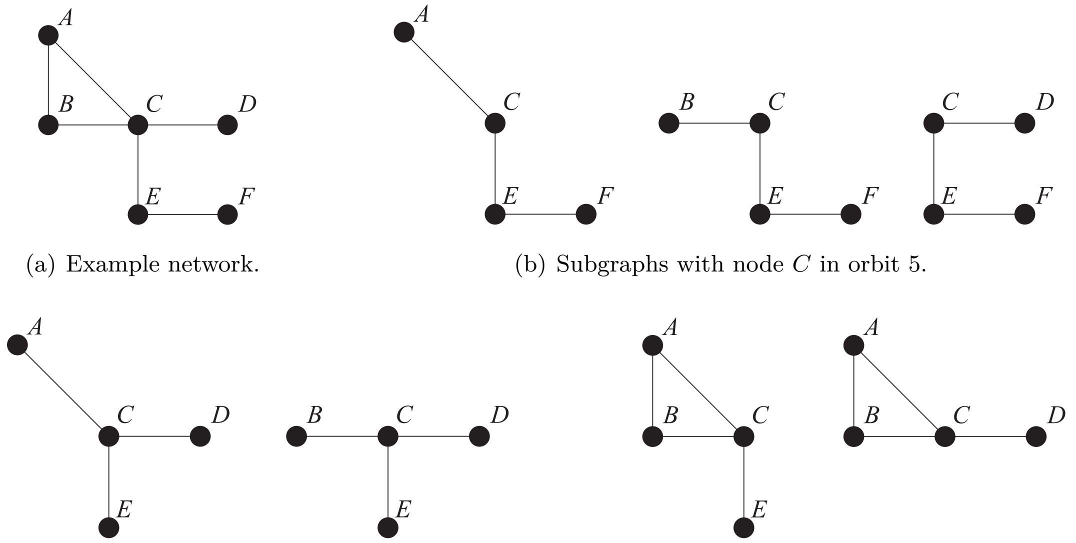
The notation used throughout the paper is summarized in Table 1.

2. Combinatorial approach to orbit counting

Let $G = (V, E)$ be a simple graph with n vertices (V) and e edges (E). We assume that the graph is sparse ($e = O(n)$). We will denote graphlets as G_i and node orbits as O_i. We follow the enumeration by Pržulj (2007) (see Figure 1(a)), in which the orbit numbers are assigned somewhat arbitrarily but with the constraint that the indices of orbits belonging to the graphlets with fewer edges are smaller than those belonging to the graphlets with more edges. We will use E_i to denote edge orbits, which we enumerate as shown in Figure 1(b). Here we decided to ignore the pre-existing enumeration by Solava *et al.* (2012) and define a more consistent one in which the edge orbits are ordered by the orbits of the corresponding nodes.

The task is to count the number of times a node x appears in each orbit O_i, or the number

[1] In general, two nodes will have the same signature for k node graphlets if their local neighborhood of up to $k - 1$ edges is the same.



(a) Example network.

(b) Subgraphs with node C in orbit 5.

(c) Subgraphs with node C in orbit 7.

(d) Subgraphs with node C in orbit 11.

Orbit	Count	
0	4	The count of orbit 0 equals the degree of the node.
1	1	The terminal node of a path on three nodes ($\{C, E, F\}$).
2	5	The middle node of a path on three nodes ($\{A, C, D\}$, $\{A, C, E\}$, $\{B, C, D\}$, $\{B, C, E\}$, $\{D, C, E\}$).
3	1	Triangle ($\{A, B, C\}$).
5	3	The inner node of a path on four nodes (Fig. b).
7	2	The central node of a 3-edge star (Fig. c).
11	2	The central node of the $L_{3,1}$ lollipop graph (Fig. d).

(e) Non-zero orbit counts for node C.

Orbit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	2	2	0	1	1	0	1	0	0	0	2	0	0	0	0
B	2	2	0	1	1	0	1	0	0	0	2	0	0	0	0
C	4	1	5	1	0	3	0	2	0	0	0	2	0	0	0
D	1	3	0	0	1	0	2	0	0	1	0	0	0	0	0
E	2	3	1	0	0	3	2	0	0	1	0	0	0	0	0
F	1	1	0	0	3	0	0	0	0	0	0	0	0	0	0

(f) Orbit counts for all nodes.

Figure 2: Illustration of orbit counts for a simple network.

of times an edge e appears in orbit E_i. We will denote the two numbers by $o_i(x)$ and $e_i(x)$; where possible, we will omit x and write only o_i and e_i. The algorithm computes the counts for all graph nodes (or edges). Computation for just a few nodes can be done faster using a brute force approach (exhaustive enumeration).

Past approaches – such as that in **GraphCrunch** (Milenković, Lai, and Pržulj 2008) and **RAGE** (rapid graphlet enumerator; Marcus and Shavitt 2012) – are based on exhaustive enumeration

$G = (V, E)$	The observed graph with nodes V and edges E.
n	Number of nodes in the graph.
e	Number of edges.
d	Maximal node degree.
k	Graphlet size.
O_i	Node orbit i.
$o_i(x)$ (or o_i)	The number of times that node x appears in orbit O_i; we use o_i to reduce the clutter where possible.
E_i	Edge orbit i.
$e_i(x)$ (or e_i)	The number of times that node x appears in orbit E_i.
$N(x_1, x_2, \ldots x_i)$	The set of common neighbors of nodes x_1, x_2, …, x_i.
$c(x_1, x_2, \ldots x_i)$	The number of common neighbors of nodes x_1, x_2, …, x_i.
$G[\{x_1, x_2, \ldots x_i\}]$	A subgraph of G on nodes x_1, x_2, …, x_i.
$\cong$	Symbol $\cong$ denotes graph isomorphism.

Table 1: Notation used in the paper.

of induced subgraphs.[2] Their theoretical and empirical complexity of enumerating graphlets of size k is $O(nd^{k-1})$, where d is the maximal node degree in the graph. Our approach builds on the work of Kloks, Kratsch, and Müller (2000), who constructed a system of equations for counting induced subgraphs with four-nodes, and Kowaluk, Lingas, and Lundell (2011), who generalized it for larger subgraphs. We use a similar principle to count orbits; besides, our approach scales better for sparse graphs. In comparison with enumeration-based algorithms, the combinatorial approach decreases the practical time complexity by the factor of d by directly enumerating only the graphlets of size $k - 1$ and using them to compute the counts for graphlets of size k.

2.1. Node orbits

We shall demonstrate the basic idea with an example.

Let x be a node in the graph G. $o_{45}(x)$ represents the number of times x appears in orbit O_{45}, that is, the number of ways in which G_{19} can be embedded in G so that x is in orbit O_{45}. To reduce the clutter, we shall omit x and denote this by o_{45}. Counts o_{56}, o_{62} and o_{65} are defined similarly. We will show that the following relation holds for any x:

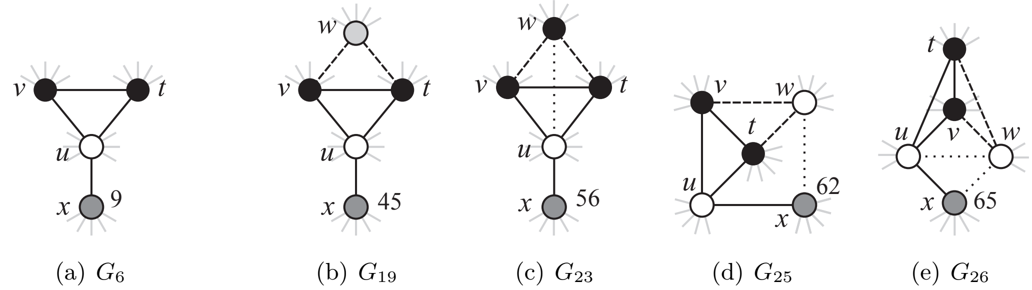
$$o_{45} + 3o_{56} + 2o_{62} + 2o_{65} = \sum_{\substack{u,v,t:\ G[\{x,u,v,t\}] \cong G_9 \\ v<t\ \wedge\ v,t \notin N(x)}} \left(c\left(v,t\right) - 1\right), \qquad (1)$$

where u, v and t are triplets of nodes that fulfill certain conditions (details are explained below) and $c(v, t)$ is the number of common neighbors of v and t. The left-hand side of the equation is a linear combination of orbit counts that we wish to compute and the right-hand side is a statistics that is easy to obtain.

Equation 1 can be constructed as follows.[3] Let the subgraph on some nodes x, u, v and t be isomorphic to G_6 with x in O_9 (Figure 3(a)). Now we observe the possible extensions of

[2] Recent versions of **GraphCrunch** also already include a part of our approach described here.

[3] This description is intended to present the reasoning behind the relations, while the actual construction was slightly different in order to obtain a useful system of equations. Details are given in Section 2.3.



(a) G_6 (b) G_{19} (c) G_{23} (d) G_{25} (e) G_{26}

Figure 3: Derivation of the relation between o_{45}, o_{56}, o_{62} and o_{65}. Solid lines belong to graphlet G_6, dashed lines represent the required edges (as described in Section 2.3) and dotted lines represent additional edges whose presence or absence determines the orbit of the node x. Gray lines represent edges to other nodes of G.

$G[\{x, u, v, t\}]$ with a node $w \in V$ that is attached to v and t. For each such $w \in N(v, t)$, the subgraph on x, u, v, t, w is isomorphic

- to G_{19}, if w is not connected to x and u (Fig. 3(b)), or
- to G_{23}, if w is connected to u, but not to x (Fig. 3(c)), or
- to G_{25}, if w is connected to x, but not to u (Fig. 3(d)), or
- to G_{26}, if w is connected to x and u (Fig. 3(e)).

This puts x in orbits O_{45}, O_{56}, O_{62} or O_{65}, respectively. Therefore,

$$o'_{45} + o'_{56} + o'_{62} + o'_{65} = |N(v, t)| - 1 = c(v, t) - 1, \tag{2}$$

where o'_i represent orbit counts considering only these particular nodes (and annotations) u, v, t, and all common neighbors of v and t, $N(v, t)$. The term -1 is needed since one of the members of $N(v, t)$ is also u.

Equation 1 relates the total orbit counts o_{45}, o_{56}, o_{62} and o_{65} for a fixed node x. We construct it by summing the right-hand side of (2), $c(v, t) - 1$, over all triplets $\{u, v, t\} \subseteq V$ such that $G[\{x, u, v, t\}] \cong G_9$ and $v, t \notin N(x)$ (to put x in O_9 within this subgraph) and with $v < t$ under some arbitrary ordering of nodes, that is,

$$\sum_{\substack{u,v,t:\ G[\{x,u,v,t\}]\cong G_9 \\ v<t\ \wedge\ v,t\notin N(x)}} (c(v, t) - 1)\,. \tag{3}$$

Despite the condition $v < t$, some subgraphs are counted multiple times.

- Each subgraph with x in O_{56} ($G[\{x, u, v, t, w\}] \cong G_{23}$, Figure 3(c)) is counted thrice: The nodes x and u are fixed while v, t and w are exchanging their roles in three possible permutations (the condition $v < t$ prohibits the other three out of the six possible permutations).

- If x belongs to O_{62} ($G[\{x, u, v, t, w\}] \cong G_{25}$, Figure 3(d)), the subgraph on quintuplet $\{x, u, v, t, w\}$ is counted twice, with exchanged roles of u and w; the nodes v and t are fixed due to $v < t$.
- The configuration in which x is in O_{65} ($G[\{x, u, v, t, w\}] \cong G_{26}$, Figure 3(e)) is similar to that of O_{62}.
- The configuration for x in O_{45} ($G[\{x, u, v, t, w\}] \cong G_{19}$, Figure 3(b)) is unique: For quintuplet $\{x, u, v, t, w\}$ in $x \in O_{45}$, the conditions $v < t$ and $v, t \notin N(x)$ allow for only one possible annotation of the nodes.

After accounting for these multiple counts of the same orbit when summing (2) over all applicable triplets u, v, t (as in (3)), we get Equation 1. To evaluate such equations we need to pre-compute values $c(v, t)$ and sum them over four-node induced subgraphs of the network. Both of these steps require an enumeration of all four-node induced subgraphs, which is the bottleneck of the method. Because every enumerated four-node subgraph will contribute to the sum on the right side of one or more equations, we can optimize the code and avoid explicitly checking the summation conditions in the equations because they are already included in the enumeration process (see the code snippet in Section 2.4 for illustration).

Certain relations involve more complicated symmetries, for instance

$$o_{37} + 2o_{68} + 2o_{64} + 2o_{63} + 4o_{62} + o_{53} + o_{51} + 4o_{49} = \sum_{\substack{u,v,t:\ G[\{x,u,v,t\}] \cong G_5 \\ u<v\ \wedge\ u,v \in N(x)}} (c(u) - 2 + c(v) - 2)\,. \quad (4)$$

The sum runs over all induced subgraphs in G that put node x in orbit O_8 in G_5. Nodes u and v are its neighbors and t is the node opposite of x in G_5. We obtain the same graphlet if we attach a new node w either to u or to v, and there are $c(u) - 2$ and $c(v) - 2$ such possibilities. There are also three optional edges (to x, y and u or v), which decide the orbit of x in the extended graphlet; the resulting orbit can be O_{37}, O_{68}, O_{64}, O_{63}, O_{62}, O_{53}, O_{51} or O_{49}.

2.2. Edge orbits

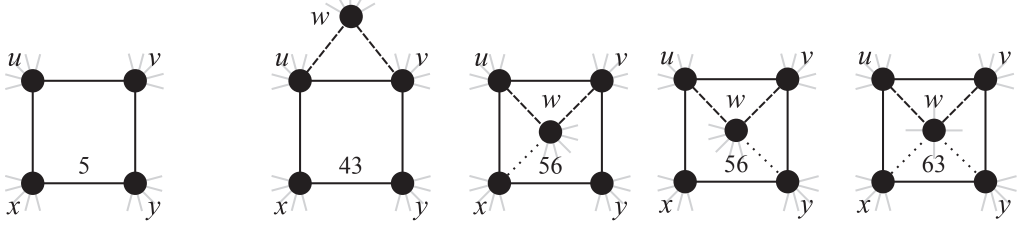
Relations for edge orbits are derived in the same way. For instance, let (x, y) represent an edge of a square (graphlet G_5). Let us label the remaining two nodes with $u \in N(x)\backslash\{y\}$ and $v \in N(y)\backslash\{x\}$ (Figure 4(a)). Extending this pattern with a node w that spans over the edge (u, v) leads to three possible graphlets and hence three different edge orbits of the edge (x, y) (Figure 4 (b–d)).

Orbit E_{56} arises when w is adjacent to either x or y, while the other two orbits, E_{43} and E_{63} can only arise in one way. Hence the relation between the orbits is

$$e_{43} + 2e_{56} + e_{63} = \sum_{\substack{u,v:\ G[\{x,y,u,v\}] \cong G_5 \\ (x,y)\in E\ \wedge\ u \in N(x)\ \wedge\ v \in N(y)}} c(u, v). \quad (5)$$

2.3. System of equations

We constructed the equations similar to those above to relate each orbit with orbits from graphlets with a larger number of edges. Complete lists of equations are provided in the appendices.



(a) G_5 (b) G_{19} (c) G_{23} with (x, w) and (y, w) (d) G_{25}

Figure 4: Derivation of the relation between e_{43}, e_{56} and e_{63}. Solid lines belong to graphlet G_5, dashed lines represent the required edges (since w is defined to span over the edge (u, v)) and dotted lines represent additional edges whose presence or absence determines the orbit of the edge (x, y). Gray lines represent edges to other nodes of G.

For instance, Equation 1 was constructed specifically to relate o_{45} with higher orbits. The actual construction of the equation goes in the opposite direction from that presented in the introductory example. We started with the graphlet G_{19}, to which the orbit O_{45} belongs and picked one of the nodes (labeled w in the above case). We assumed that w is adjacent to v and t, and examined the graphlets in which w may be also adjacent to u and/or x. This ensures that the equation that is set up with O_{45} in mind relates O_{45} with orbits with higher indices since these graphlets have more edges than G_{19}. As a consequence, the resulting system of equations is triangular, and thus independent and easy to solve by going backwards from the higher orbits (starting with O_{14} or O_{72}, which belong to complete graphs) towards lower orbits.

We impose several constraints on selection of w. Node w cannot coincide with x, or with x or y when computing orbits of edge (x, y). We further require that removal of w does not break the remaining nodes into disconnected subgraphs. Node w must have at most $k - 2$ neighbors; when it does have $k - 2$ neighbors, they must be connected. This allows for more time- and space-efficient computations of orbits, as described in Section 2.4. Existence of such nodes for each orbit of four- and five-node graphlets can be proven by exhaustive search, with exception of G_5, which is handled as a special case.

All equations have the following general form:

$$a_1 o_{i_1} + a_2 o_{i_2} + ... + a_t o_{i_t} = \sum_{\substack{\mathcal{S}:G[\mathcal{S}]\cong G_j \\ x\in\mathcal{S} \,\wedge\, \mathrm{cond}(\mathcal{S})}} \left(c(\mathcal{S}_1) + c(\mathcal{S}_2) + \ldots + c(\mathcal{S}_u) + C \right), \qquad (6)$$

where $\mathrm{cond}(\mathcal{S})$ is a set of conditions that constrain the embedding of $G[\mathcal{S}]$ into G and assign labels to nodes. For instance, in Equation 1, condition $v, t \notin N(x)$ assigns the labels v and t to the nodes in orbit O_{10} and $v < t$ ensures that the same quadruplet of nodes is not counted twice.

The sum runs over subgraphs $G[\mathcal{S}]$ isomorphic to some graphlet G_j on $k - 1$ nodes, that is, over some three-node graphlet when computing the orbits in four-node graphlets, or over some four-node graphlet when computing orbits in five-node graphlets. The subgraph must include x, and the conditions in the sum put x into some fixed orbit. Additional conditions may impose ordering on the remaining nodes of the graphlet to decrease the number of symmetries.

The terms in the sum are the number of common neighbors of some subsets of nodes in the subgraph ($\mathcal{S}_k \subset \mathcal{S}$). The number of such terms is between 1 and 3. The size of $\mathcal{S}_k$ is also between 1–3, that is, the terms refer to node degrees and to the number of common neighbors of pairs and triplets of nodes. The criteria for the choice of node w, which are described above, ensure that these terms can be efficiently computed using some pre-computed data as described in the following section.

The left-hand side is a fixed linear combination of orbits to which the node x evolves after extending G_j with another node connected to one of subsets $\mathcal{S}_k$. The coefficients reflect the symmetries in the graphlets with regard to node assignments.

We prepared a system of 10 equations that relate the 11 node orbits of four-node graphlets, and a system of 57 equations that relate the 58 node orbits of the five-node graphlets. Likewise, we have constructed 9 equations that relate the 10 edge orbits for four-node graphlets and 55 equations for 56 edge orbits on five-node graphlets. By selecting different nodes w, we have empirically verified that it is impossible to construct a full-rank system using our approach and the constraints we put on w.

Due to the rank's deficiency, one of the orbits must be enumerated directly. The most suitable candidates are the orbits belonging to complete graphlets (O_{14} and O_{72} for nodes, and E_{12} and E_{68} for edges). First, this allows for a straightforward computation of the orbits since the system is triangular so that lower orbits are computed from the higher. Second, since we assume that the graphs are sparse, we can efficiently compute these orbits by using an enumeration method similar to the Bron-Kerbosch maximal clique enumeration algorithm (Bron and Kerbosch 1973).

2.4. Algorithm

The algorithm consists of pre-computation of some data, followed by computation of orbit counts for each node or edge.

1. Pre-computation:
    - Count the complete graphlets touched by each node or edge.
    - Count the common neighbors of each pair and each connected triplet of vertices.
2. For each node or edge:
    - Compute the right-hand sides by enumeration of $k-1$ node graphs using the pre-computed data above.
    - Solve the system of linear equations.

Our implementation of the algorithm represents the graph with adjacency and incidence lists, which are appropriate for sparse graphs. If the graph has less than 30000 nodes, we also construct an adjacency matrix. The matrix, which uses 1 bit per edge and takes at most around 100 MB, allows us to check for existence of edges between any given pair of nodes in constant time. Without it, the time complexity of the look-up for an edge between two nodes is proportional to the logarithm of the number of neighbors of one of the nodes.

In the following, we will describe each step in more detail.

1. Pre-computation:

   - For each node, count the number of complete graphlets in which the node or edge participates. We build cliques of size k from cliques of size $k-1$ by maintaining a set of candidate nodes that are adjacent to all nodes in the smaller clique. This procedure is similar to the Bron-Kerbosch algorithm with the difference that we are not interested in maximal cliques but in all cliques of a given size.

     Although the theoretical upper bound of the time complexity of this step is $O(ed^{k-2})$, where d is the maximal node degree, the actual contribution of this step to the total running time is negligible since complete subgraphs (cliques) in sparse networks are rare.

   - Compute and store the number of common neighbors for each pair of adjacent vertices. This takes $O(ed)$ time and $O(e)$ space. For computing the orbits in five-node graphlet, we also compute the number of paths of length 2 between each pair of nodes for which such a path exists, and the number of common neighbors for all triplets of connected nodes. This takes $O(ed^2)$ time and $O(ed)$ space.

2. For each node or edge:

   - Compute the right-hand sides of the system of the linear equations. Its general form is shown in Equation 6. For four-node graphlets, the sums run over three-node paths or triangles in which the node appears. For five-node graphlets, they run over four-node graphlets that the node touches.

     Right-hand sides of equations that sum over the same graphlet can be computed simultaneously. The following code chunk illustrates the computation of the right-hand sides of equations for orbits 13, 16 and 20 for an edge (x, y),

$$e_{13} + 2e_{22} + 2e_{28} + e_{31} + e_{40} + 2e_{44} + 2e_{54} = \sum_{\substack{a,b\colon G[\{x,y,a,b\}]\cong G_3 \\ a\in N(x)\ \wedge\ b\in N(y)}} (c(a) + c(b) - 2),$$

$$2e_{16} + 2e_{20} + 2e_{22} + e_{31} + 2e_{40} + e_{44} + 2e_{54} = \sum_{\substack{a,b\colon G[\{x,y,a,b\}]\cong G_3 \\ a\in N(x)\ \wedge\ b\in N(y)}} (c(x) + c(y) - 4),$$

$$e_{20} + e_{40} + e_{54} = \sum_{\substack{a,b\colon G[\{x,y,a,b\}]\cong G_3 \\ a\in N(x)\ \wedge\ b\in N(y)}} c(x, y).$$

The code for computation of the right-hand sides is as follows.

```
for (int nx = 0; nx < deg[x]; nx++) {
  int const &a = adj[x][nx];
  if (a == y || adjacent(y, a))
    continue;
  for (int ny = 0; ny < deg[y]; ny++) {
    int const &b = adj[y][ny];
    if (b == x || adjacent(x,b) || adjacent(a,b))
       continue;
    EORBIT(3)++;
    f_13 += (deg[a] - 1) + (deg[b] - 1);
    f_16 += (deg[x] - 2) + (deg[y] - 2);
    f_20 += tri[xy];
  }
}
```

Here, `deg[x]` and `deg[y]` are degrees of nodes x and y, and `adj[x]` and `adj[y]` are arrays with indices of their neighbors. Function `adjacent(u, t)` checks whether nodes u and t are adjacent (with a time complexity $O(1)$ or $O(\log d)$, depending on whether we construct an adjacency matrix or not) and `tri[xy]` is the number of triangles spanning over the edge between x and y. Variables `f_13`, `f_16` and `f_20` contain the right-hand sides of equations for O_{13}, O_{16} and O_{20}.

The `if`-clauses check that the edge belongs to E_3 and impose additional constraints as needed. The computation is sped up by using the pre-computed data from the first two steps. In the above case, the right-hand sides of equations for orbits 13, 16 and 20 are $(c(a)-1)+(c(b)-1)$, $(c(x)-2)+(c(y)-2)$ and $c(x,y)$, respectively. The former two are trivial to compute from the graph, and the latter is pre-computed in the second step above.

Note that the orbits for $k-1$-node graphlets (as the orbit 3, above) are computed directly.

The time complexity of this step is $O(ed^{k-3})$.

- Solve the system of equations to obtain orbit counts. Since the system is triangular and the coefficients are fixed, this does not require decomposing or inverting a matrix; the orbits are computed in order, from the higher towards the lower indices, starting with the orbit belonging to the complete graphlet, as for instance, in the following code snippet from the computation of edge orbits.

```
EORBIT(67) = C5[e];
EORBIT(66) = (f_66 - 6 * EORBIT(67)) / 2;
EORBIT(65) = (f_65 - 6 * EORBIT(67));
EORBIT(64) = (f_64 - 2 * EORBIT(66));
EORBIT(63) = (f_63 - 2 * EORBIT(65)) / 2;
EORBIT(62) = (f_62 - 2 * EORBIT(66) - 3 * EORBIT(67));
EORBIT(61) = (f_61 - 2 * EORBIT(65) - 4 * EORBIT(66)
  - 12 * EORBIT(67));
EORBIT(60) = (f_60 - 1 * EORBIT(65) - 3 * EORBIT(67));
```

The system of equations is also rather sparse, with each equation having at most (but usually much less than) eight variables. These nice properties – sparse triangular shape – do not make the algorithm faster since the coefficients are fixed. Even a more general matrix could be inverted in advance and hard-coded into the program. The advantage of triangularity, besides the interpretability of the program, is the numerical accuracy since the entire computation stays in the realm of whole numbers.[4]

The system is solved once for each node (or edge), so the time complexity is $O(n)$ ($O(e)$ for edge-orbits).

[4] The implementation of the R package uses 64-bit integers internally, but returns a matrix of double precision numbers since some orbit counts for larger graphs do not fit into 32-bit integers used in R.

The total time complexity for all four steps is $O(ed^{k-2} + ed^{k-3} + ed^{k-3} + n)$ for nodes and $O(ed^{k-2} + ed^{k-3} + ed^{k-3} + e)$ for edges. The theoretical complexity is thus $O(ed^{k-2})$, which is the same as for direct enumeration algorithms. Since large networks are typically sparse, the actual contribution of the first term, which comes from enumerating the cliques with k nodes, is negligible in practice. Empirical measurements indeed show that the time complexity is proportional to ed^{k-3}, that is, $O(ed)$ for four-node graphlets and $O(ed^2)$ for five-node graphlets.

3. The orca package

Package **orca** (orbit counter) is written mostly in C++, with coercion and wrapper functions in R. The package requires R version 2.15 or higher. The package is available from the Comprehensive R Archive Network (CRAN) at https://CRAN.R-project.org/package=orca (Hočevar and Demšar 2016). Due to using the C++ standard 11, which is not available on all platforms, CRAN only hosts the package for R 3.1. Packages for R 2.15 and binaries for OS X and MS Windows are available on the supplement page (http://www.biolab.si/supp/Rorca/).

3.1. Functions

The package provides four functions: `count4` and `count5` count the node orbits of graphlets on up to four and up to five nodes, and `ecount4` and `ecount5` count the edge orbits. All functions accept a single argument, a graph stored in

- a graph object from the **graph** package (Gentleman, Whalen, Huber, and Falcon 2016);
- an $e \times 2$ edge matrix in which each row contains a pair of nodes given by one-based integer indices; or
- a data frame in the same format.

Functions return a numeric matrix with rows corresponding to graph nodes or edges, and the columns corresponding to orbits, with column 1 corresponding to orbit 0, column 2 to orbit 1 and so forth.[5]

We will show the package usage on the Karate club network (Zachary 1977), which is included in the package. The network is visualized in Figure 5.

```
R> library("orca")
R> data("karate", package = "orca")
R> dim(karate)

[1] 78  2

R> max(karate)

[1] 34
```

[5] In the paper we adhere to the traditional numbering of orbits, which starts with 0, to avoid confusion. In practice, the numbering seldom matters since we typically observe the differences between orbit signatures of nodes and edges, in which we consider the whole vectors and not individual orbits.

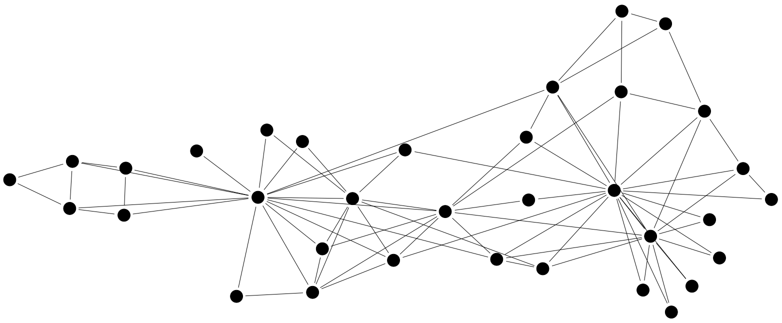


Figure 5: Karate club network.

The network has 78 edges (the number of rows of the matrix) and 34 nodes (the maximal node index in the matrix).

```
R> orbits <- count4(karate)
R> dim(orbits)
```

```
[1] 34 15
```

The result of `count4`, which counts node orbits for graphlets with up to four nodes, has 34 rows (the number of nodes) and 15 columns (the number of orbits).

The first four orbits correspond to three-node graphlets. Here are the orbit counts of four-node graphlets for the first four nodes.

```
R> orbits[1:4, 5:15]
```

```
     O4  O5  O6  O7  O8  O9 O10 O11 O12  O13  O14
[1,] 81 197  13 352  10   6  34 171   2   30    7
[2,] 73  56  33  32   6   8  80  27   2   18    7
[3,] 72 179  84  54  20  17  75  51   6    8    7
[4,] 49  11  56   1   0   5  81   5   4    7    7
```

Note that for such small networks a visualization reveals more than orbit counts. Orbit counts become useful on large networks, which are difficult to plot out.

3.2. Usage example on the Wikipedia for Schools network

Thiel and Berthold (2012) argue that in exploring networks we are not necessarily interested in nodes that are closely positioned to the query node (spatial similarity), but also in nodes that have a similar neighborhood structure (structural similarity). They proposed activation spreading signature as a topological description of the local neighborhood of graph vertices and demonstrate its use on the Wikipedia for Schools network. We conducted a similar experiment by using orbit counts instead of activation spreading for the signature.

We downloaded the 2013 edition of Wikipedia for Schools (SOS Children 2013) and extracted the network of internal links.[6] We computed the orbits for four-node graphlets and found

[6] The network is available for download at http://www.biolab.si/supp/Rorca/.

the nearest neighbors (in terms of Euclidean distances between orbit counts) using **FNN** (Beygelzimer, Kakadet, Langford, Arya, Mount, and Li 2013) for a few nodes.

```
R> library("orca")
R> library("FNN")
R> nodes <- scan("schools-wiki-nodes.txt", what = "", sep = "\n")
R> edges <- read.table("schools-wiki-edges.txt")
R> orbits <- count4(edges)
R> nn <- get.knn(orbits, k = 10)
R> neighbors <- nn$nn.index
R> distances <- nn$nn.dist
R> check <- c("Canada", "Germany", "Isaac Newton", "Albert Einstein",
+    "Mahatma Gandhi", "Mahabharata")
R> node_indices <- match(check, nodes)
R> for (i in 1:length(check)) {
+    cat("\n\n", check[i], ": ", sep = "")
+    s <- mapply(function(x, y) sprintf("%s (%i)", x, y),
+      nodes[neighbors[node_indices[i], ]],
+      round(distances[node_indices[i], ] / 1000))
+    cat(s, sep = ", ")
+  }
```

Computation of orbits for 4-node graphlets takes 6.1 seconds on a desktop computer. In comparison, **GraphCrunch** as currently the fastest pure enumeration approach[7] takes 13.8 minutes. Computation of 5-node orbits takes 115 minutes; **GraphCrunch** needs 249 hours. This represents a speed-up by a factor of about 130.

After the nodes are described by orbit counts, we find the ten most similar nodes (as defined by Euclidean distance, for the sake of simplicity) to several selected nodes. Results, together with distances divided by 1000, are as follows.

```
Canada: Japan (3548), Italy (4224), Russia (6962), Africa (15546),
Spain (15963), London (18186), Australia (18356), Latin (19360),
China (20146), 19th century (26147)

Germany: India (3384), World War II (7343), China (16753), Australia (18652),
London (19340), Italy (32870), Europe (35056), Canada (36875), Japan (40001),
Russia (43656)

Isaac Newton: Temperature (252), Church of England (297), Jupiter (420),
University of Cambridge (432), Planet (441), Science (518),
Albert Einstein (519), Evolution (545), Elephant (548), Insect (597)

Albert Einstein: Science (165), Climate change (249), Charles Darwin (267),
Jupiter (332), Celsius (366), United Kingdom of Great Britain and Ireland
(415), Church of England (435), United States Congress (452),
```

[7] Newer versions of **GraphCrunch** also already include some parts of the algorithm described here.

```
Black Sea (471), Civilization (501)

Mahatma Gandhi: Oil refinery (97), Friedrich Engels (98), Oil shale (100),
Impressionism (103), Rugby league (103), Tropic of Cancer (111),
Reggae (111), Non-governmental organization (125),
John Maynard Keynes (128), John Stuart Mill (135)

Mahabharata: Feather (38), Shiva (38), Guitar (62), John Vanbrugh (66),
Fever (66), Introduction to evolution (68), Henry IV of England (69),
Microscope (69), René Descartes (71), 1754 (72)
```

The results for *Canada* and *Germany* are impressive. The two nodes have similar orbit counts – and thus a similar role in the local network topology – as nodes *Japan*, *Italy*, *Russia* and other nodes representing countries, cities and regions. This would indicate that it is possible to recognize the nodes corresponding to countries based on the local network structure represented by orbit counts.

The node orbits – and thus the structure of the network around them – for *Isaac Newton* and *Albert Einstein* are also similar to those of other nodes related to physics. The inclusion of *World War II* in the nodes similar to *Germany*, and the *Church of England* with *Isaac Newton* may, however, be just an instance of the Texas sharpshooter phenomenon. Results for *Mahatma Gandhi* and *Mahabharata* are considerably less satisfactory as these two nodes are connected to unrelated nodes.

Exploring why the topology around the nodes is similar in one case and not in another is beyond the scope of this paper.[8] While this example provides an alternative take at the problem explored by Thiel and Berthold (2012), graphlet analysis is most often used in bioinformatics, where orbit counts are assumed to reflect the roles of genes or proteins in the observed networks. An interested reader may find further examples in the cited works of Pržulj and Milenković.

4. Conclusion

We presented a new package **orca** for computing the graphlet orbit counts for nodes and edges. This paper provides the first complete description of the underlying algorithm, which runs much faster than the previous approaches; a more detailed comparison is available in Hočevar and Demšar (2013). The novel contribution of the paper is also the generalization of the method to counting the orbits for edges. The package is available on the CRAN repository under the GPL-3 license.

Acknowledgments

This work has been funded by the Slovenian research agency grants J2-5480 and P2-0209.

[8] We speculate that nodes that represent entities of the same kind, such as countries, are connected, thus the local topology is similar because the nodes are actually close to each other. Yet this does not explain why the method fails for Mahatma Gandhi and Mahabharata for which it found some very similar but unrelated neighbors. A better explanation might require investigating how this network has been constructed.

References

Beygelzimer A, Kakadet S, Langford J, Arya S, Mount D, Li S (2013). ***FNN**: Fast Nearest Neighbor Search Algorithms and Applications.* R package version 1.1, URL https://CRAN.R-project.org/package=FNN.

Bron C, Kerbosch J (1973). "Algorithm 457: Finding All Cliques of an Undirected Graph." *Communications of the ACM*, **16**(9), 575–577. doi:10.1145/362342.362367.

Gentleman R, Whalen E, Huber W, Falcon S (2016). ***graph**: A Package to Handle Graph Data Structures.* R package version 1.50.0, URL http://www.Bioconductor.org/packages/release/bioc/html/graph.html.

Hočevar T, Demšar J (2013). "A Combinatorial Approach to Graphlet Counting." *Bioinformatics*, **30**(4), 559–565. doi:10.1093/bioinformatics/btt717.

Hočevar T, Demšar J (2016). ***orca**: Computation of Graphlet Orbit Counts in Sparse Graphs.* R package version 1.1-1, URL https://CRAN.R-project.org/package=orca.

Kloks T, Kratsch D, Müller H (2000). "Finding and Counting Small Induced Subgraphs Efficiently." *Information Processing Letters*, **74**(3–4), 115–121. doi:10.1016/s0020-0190(00)00047-8.

Kowaluk M, Lingas A, Lundell EM (2011). "Counting and Detecting Small Subgraphs via Equations and Matrix Multiplication." In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1468–1476. doi:10.1137/110859798.

Marcus D, Shavitt Y (2012). "**RAGE** – A Rapid Graphlet Enumerator for Large Networks." *Computer Networks*, **56**(2), 810–819. doi:10.1016/j.comnet.2011.08.019.

Milenković T, Lai J, Pržulj N (2008). "**GraphCrunch**: A Tool for Large Network Analyses." *BMC Bioinformatics*, **9**, 70. doi:10.1186/1471-2105-9-70.

Pržulj N (2007). "Biological Network Comparison Using Graphlet Degree Distribution." *Bioinformatics*, **23**(2), 177–183. doi:10.1093/bioinformatics/btl301.

Pržulj N, Corneil D, Jurisica I (2004). "Modeling Interactome: Scale-Free or Geometric?" *Bioinformatics*, **20**(18), 3508–3515. doi:10.1093/bioinformatics/bth436.

Solava R, Michaels R, Milenković T (2012). "Graphlet-Based Edge Clustering Reveals Pathogen-Interacting Proteins." *Bioinformatics*, **28**(18), 480–486. doi:10.1093/bioinformatics/bts376.

SOS Children (2013). "Wikipedia for Schools." URL http://schools-wikipedia.org/.

Thiel K, Berthold M (2012). "Node Similarities From Spreading Activation." In M Berthold (ed.), *Bisociative Knowledge Discovery*, volume 7250 of *Lecture Notes in Computer Science*, pp. 246–262. Springer-Verlag. doi:10.1007/978-3-642-31830-6_17.

Zachary W (1977). "An Information Flow Model for Conflict and Fission in Small Groups." *Journal of Anthropological Research*, **33**(4), 452–473. doi:10.1086/jar.33.4.3629752.

A. Equations for node-orbit counts in 4-graphlets

Let $p(u, v)$ denote the number of paths on three nodes that start at node u, continue with v and end with some node t, which is not connected to u. We can compute $p(u, v)$ as $p(u, v) = deg(v) - 1 - c(u, v)$.

$$\begin{aligned}
o_{12} + 3o_{14} &= \sum_{y,z\colon y<z, G[\{x,y,z\}]\cong G_2} c(y, z) - 1 \\
2o_{13} + 6o_{14} &= \sum_{y,z\colon y<z, G[\{x,y,z\}]\cong G_2} (c(x, y) - 1) + (c(x, z) - 1) \\
o_{10} + 2o_{13} &= \sum_{y,z\colon y<z, G[\{x,y,z\}]\cong G_2} p(y, z) + p(z, y) \\
2o_{11} + 2o_{13} &= \sum_{y,z\colon y<z, G[\{x,y,z\}]\cong G_2} p(y, x) + p(z, x) \\
6o_7 + 2o_{11} &= \sum_{y,z\colon y<z, y,z\in N(x), G[\{x,y,z\}]\cong G_1} (p(y, x) - 1) + (p(z, x) - 1) \\
o_5 + 2o_8 &= \sum_{y,z\colon y<z, y,z\in N(x), G[\{x,y,z\}]\cong G_1} p(x, y) + p(x, z) \\
2o_6 + 2o_9 &= \sum_{y,z\colon x,z\in N(y), G[\{x,y,z\}]\cong G_1} p(x, y) - 1 \\
2o_9 + 2o_{12} &= \sum_{y,z\colon x,z\in N(y), G[\{x,y,z\}]\cong G_1} c(y, z) \\
o_4 + 2o_8 &= \sum_{y,z\colon x,z\in N(y), G[\{x,y,z\}]\cong G_1} p(y, z) \\
2o_8 + 2o_{12} &= \sum_{y,z\colon x,z\in N(y), G[\{x,y,z\}]\cong G_1} c(x, z) - 1
\end{aligned}$$

B. Equations for edge-orbit counts in 4-graphlets

$$\begin{aligned}
2e_{10} + 2e_{11} &= \sum_{z\colon G[\{x,y,z\}]\cong G_2} (c(x, y) - 1) \\
e_9 + 4e_{11} &= \sum_{z\colon G[\{x,y,z\}]\cong G_2} (c(x, z) + c(y, z) - 2) \\
e_8 + e_9 + 4e_{10} + 4e_{11} &= \sum_{z\colon G[\{x,y,z\}]\cong G_2} (c(x) + c(y) - 4) \\
e_7 + e_9 + 2e_{11} &= \sum_{z\colon G[\{x,y,z\}]\cong G_2} (c(z) - 2) \\
2e_6 + e_9 &= \sum_{z\colon z\in N(y)\wedge G[\{x,y,z\}]\cong G_1} c(y, z) + \sum_{z\colon z\in N(x)\wedge G[\{x,y,z\}]\cong G_1} c(x, z) \\
2e_5 + e_9 &= \sum_{z\colon z\in N(y)\wedge G[\{x,y,z\}]\cong G_1} (c(x, z) - 1) + \sum_{z\colon z\in N(x)\wedge G[\{x,y,z\}]\cong G_1} (c(y, z) - 1)
\end{aligned}$$

$$2e_4 + 2e_6 + e_8 + e_9 = \sum_{z\colon z\in N(y)\wedge G[\{x,y,z\}]\cong G_1} (c(y)-2) + \sum_{z\colon z\in N(x)\wedge G[\{x,y,z\}]\cong G_1} (c(x)-2)$$
$$2e_3 + 2e_5 + e_8 + e_9 = \sum_{z\colon z\in N(y)\wedge G[\{x,y,z\}]\cong G_1} (c(x)-1) + \sum_{z\colon z\in N(x)\wedge G[\{x,y,z\}]\cong G_1} (c(y)-1)$$
$$e_2 + 2e_5 + 2e_6 + e_9 = \sum_{z\colon z\in N(y)\cup N(x)\wedge G[\{x,y,z\}]\cong G_1} (c(z)-1)$$

C. Equations for node-orbit counts in 5-graphlets

Conditions, P_i, define the order of nodes and put x in orbit O_i; e.g., in P_{13} node x is in orbit O_{13}.

$$\begin{aligned}
P_{14}(x,u,v,t) &= u<v<t \wedge G[\{x,u,v,t\}] \cong G_8\\
P_{13}(x,u,v,t) &= v<t \wedge (v,t)\notin E \wedge G[\{x,u,v,t\}] \cong G_7\\
P_{12}(x,u,v,t) &= u<v \wedge (x,t)\notin E \wedge G[\{x,u,v,t\}] \cong G_7\\
P_{11}(x,u,v,t) &= u<v \wedge u,v\notin N(t) \wedge G[\{x,u,v,t\}] \cong G_6\\
P_{10}(x,u,v,t) &= x,u\notin N(t) \wedge G[\{x,u,v,t\}] \cong G_6\\
P_{9}(x,u,v,t) &= v<t \wedge v,t\notin N(x) \wedge G[\{x,u,v,t\}] \cong G_6\\
P_{8}(x,u,v,t) &= u<v \wedge u,v\in N(x) \wedge G[\{x,u,v,t\}] \cong G_5\\
P_{7}(x,u,v,t) &= u<v<t \wedge u,v,t\in N(x) \wedge G[\{x,u,v,t\}] \cong G_4\\
P_{6}(x,u,v,t) &= v<t \wedge x,v,t\in N(u) \wedge G[\{x,u,v,t\}] \cong G_4\\
P_{5}(x,u,v,t) &= u,v\in N(x) \wedge t\in N(v) \wedge G[\{x,u,v,t\}] \cong G_3\\
P_{4}(x,u,v,t) &= x,v\in N(u) \wedge t\in N(v) \wedge G[\{x,u,v,t\}] \cong G_3
\end{aligned}$$

Equations:

$$2o_{71} + 12o_{72} = \sum_{u,v,t\colon P_{14}(x,u,v,t)} (c(x,u,v)-1) + (c(x,u,t)-1) + (c(x,v,t)-1)$$
$$o_{70} + 4o_{72} = \sum_{u,v,t\colon P_{14}(x,u,v,t)} c(u,v,t)-1$$
$$4o_{69} + 2o_{71} = \sum_{u,v,t\colon P_{13}(x,u,v,t)} c(x,v,t)-1$$
$$o_{68} + 2o_{71} = \sum_{u,v,t\colon P_{13}(x,u,v,t)} c(u,v,t)-1$$
$$o_{67} + 12o_{72} + 4o_{71} = \sum_{u,v,t\colon P_{14}(x,u,v,t)} (c(x,u)-2) + (c(x,v)-2) + (c(x,t)-2)$$
$$o_{66} + 12o_{72} + 2o_{71} + 3o_{70} = \sum_{u,v,t\colon P_{14}(x,u,v,t)} (c(u,v)-2) + (c(u,t)-2) + (c(v,t)-2)$$
$$2o_{65} + 3o_{70} = \sum_{u,v,t\colon P_{12}(x,u,v,t)} c(u,v,t)$$
$$o_{64} + 2o_{71} + 4o_{69} + o_{68} = \sum_{u,v,t\colon P_{13}(x,u,v,t)} c(v,t)-2$$

$$o_{63} + 3o_{70} + 2o_{68} = \sum_{u,v,t\colon P_{12}(x,u,v,t)} c(x,t) - 2$$

$$2o_{62} + o_{68} = \sum_{u,v,t\colon P_{8}(x,u,v,t)} c(u,v,t)$$

$$2o_{61} + 4o_{71} + 8o_{69} + 2o_{67} = \sum_{u,v,t\colon P_{13}(x,u,v,t)} (c(x,v) - 1) + (c(x,t) - 1)$$

$$o_{60} + 4o_{71} + 2o_{68} + 2o_{67} = \sum_{u,v,t\colon P_{13}(x,u,v,t)} (c(u,v) - 1) + (c(u,t) - 1)$$

$$o_{59} + 6o_{70} + 2o_{68} + 4o_{65} = \sum_{u,v,t\colon P_{12}(x,u,v,t)} (c(u,t) - 1) + (c(v,t) - 1)$$

$$o_{58} + 4o_{72} + 2o_{71} + o_{67} = \sum_{u,v,t\colon P_{14}(x,u,v,t)} c(x) - 3$$

$$o_{57} + 12o_{72} + 4o_{71} + 3o_{70} + o_{67} + 2o_{66} = \sum_{u,v,t\colon P_{14}(x,u,v,t)} (c(u) - 3) + (c(v) - 3) + (c(t) - 3)$$

$$3o_{56} + 2o_{65} = \sum_{u,v,t\colon P_{9}(x,u,v,t)} c(u,v,t)$$

$$3o_{55} + 2o_{71} + 2o_{67} = \sum_{u,v,t\colon P_{13}(x,u,v,t)} c(x,u) - 2$$

$$2o_{54} + 3o_{70} + o_{66} + 2o_{65} = \sum_{u,v,t\colon P_{12}(x,u,v,t)} c(u,v) - 2$$

$$o_{53} + 2o_{68} + 2o_{64} + 2o_{63} = \sum_{u,v,t\colon P_{8}(x,u,v,t)} c(x,u) + c(x,v)$$

$$2o_{52} + 2o_{66} + 2o_{64} + o_{59} = \sum_{u,v,t\colon P_{10}(x,u,v,t)} c(u,t) - 1$$

$$o_{51} + 2o_{68} + 2o_{63} + 4o_{62} = \sum_{u,v,t\colon P_{8}(x,u,v,t)} c(u,t) + c(t,v)$$

$$3o_{50} + o_{68} + 2o_{63} = \sum_{u,v,t\colon P_{8}(x,u,v,t)} c(x,t) - 2$$

$$2o_{49} + o_{68} + o_{64} + 2o_{62} = \sum_{u,v,t\colon P_{8}(x,u,v,t)} c(u,v) - 2$$

$$o_{48} + 4o_{71} + 8o_{69} + 2o_{68} + 2o_{67} + 2o_{64} + 2o_{61} + o_{60} = \sum_{u,v,t\colon P_{13}(x,u,v,t)} (c(v) - 2) + (c(t) - 2)$$

$$o_{47} + 3o_{70} + 2o_{68} + o_{66} + o_{63} + o_{60} = \sum_{u,v,t\colon P_{12}(x,u,v,t)} c(x) - 2$$

$$o_{46} + 3o_{70} + 2o_{68} + 2o_{65} + o_{63} + o_{59} = \sum_{u,v,t\colon P_{12}(x,u,v,t)} c(t) - 2$$

$$o_{45} + 2o_{65} + 2o_{62} + 3o_{56} = \sum_{u,v,t\colon P_{9}(x,u,v,t)} c(v,t) - 1$$

$$4o_{44} + o_{67} + 2o_{61} = \sum_{u,v,t\colon P_{11}(x,u,v,t)} c(x,t)$$

$$2o_{43} + 2o_{66} + o_{60} + o_{59} = \sum_{u,v,t\colon P_{10}(x,u,v,t)} c(v,t)$$

$$o_{42} + 2o_{71} + 4o_{69} + 2o_{67} + 2o_{61} + 3o_{55} = \sum_{u,v,t\colon P_{13}(x,u,v,t)} c(x) - 3$$

$$o_{41} + 2o_{71} + o_{68} + 2o_{67} + o_{60} + 3o_{55} = \sum_{u,v,t\colon P_{13}(x,u,v,t)} c(u) - 3$$

$$o_{40} + 6o_{70} + 2o_{68} + 2o_{66} + 4o_{65} + o_{60} + o_{59} + 4o_{54} = \sum_{u,v,t\colon P_{12}(x,u,v,t)} (c(u) - 3) + (c(v) - 3)$$

$$2o_{39} + 4o_{65} + o_{59} + 6o_{56} = \sum_{u,v,t\colon P_{9}(x,u,v,t)} (c(u,v) - 1) + (c(u,t) - 1)$$

$$o_{38} + o_{68} + o_{64} + 2o_{63} + o_{53} + 3o_{50} = \sum_{u,v,t\colon P_{8}(x,u,v,t)} c(x) - 2$$

$$o_{37} + 2o_{68} + 2o_{64} + 2o_{63} + 4o_{62} + o_{53} + o_{51} + 4o_{49} = \sum_{u,v,t\colon P_{8}(x,u,v,t)} (c(u) - 2) + (c(v) - 2)$$

$$o_{36} + o_{68} + 2o_{63} + 2o_{62} + o_{51} + 3o_{50} = \sum_{u,v,t\colon P_{8}(x,u,v,t)} c(t) - 2$$

$$2o_{35} + o_{59} + 2o_{52} + 2o_{45} = \sum_{u,v,t\colon P_{4}(x,u,v,t)} c(u,t) - 1$$

$$2o_{34} + o_{59} + 2o_{52} + o_{51} = \sum_{u,v,t\colon P_{4}(x,u,v,t)} c(x,t)$$

$$2o_{33} + o_{67} + 2o_{61} + 3o_{58} + 4o_{44} + 2o_{42} = \sum_{u,v,t\colon P_{11}(x,u,v,t)} c(x) - 3$$

$$2o_{32} + 2o_{66} + o_{60} + o_{59} + 2o_{57} + 2o_{43} + 2o_{41} + o_{40} = \sum_{u,v,t\colon P_{10}(x,u,v,t)} c(v) - 3$$

$$o_{31} + 2o_{65} + o_{59} + 3o_{56} + o_{43} + 2o_{39} = \sum_{u,v,t\colon P_{9}(x,u,v,t)} c(u) - 3$$

$$o_{30} + o_{67} + o_{63} + 2o_{61} + o_{53} + 4o_{44} = \sum_{u,v,t\colon P_{11}(x,u,v,t)} c(t) - 1$$

$$o_{29} + 2o_{66} + 2o_{64} + o_{60} + o_{59} + o_{53} + 2o_{52} + 2o_{43} = \sum_{u,v,t\colon P_{10}(x,u,v,t)} c(t) - 1$$

$$o_{28} + 2o_{65} + 2o_{62} + o_{59} + o_{51} + o_{43} = \sum_{u,v,t\colon P_{9}(x,u,v,t)} c(x) - 1$$

$$2o_{27} + o_{59} + o_{51} + 2o_{45} = \sum_{u,v,t\colon P_{4}(x,u,v,t)} c(v,t)$$

$$o_{26} + 2o_{67} + 2o_{63} + 2o_{61} + 6o_{58} + o_{53} + 2o_{47} + 2o_{42} = \sum_{u,v,t\colon P_{11}(x,u,v,t)} (c(u) - 2) + (c(v) - 2)$$

$$2o_{25} + 2o_{66} + 2o_{64} + o_{59} + 2o_{57} + 2o_{52} + o_{48} + o_{40} = \sum_{u,v,t\colon P_{10}(x,u,v,t)} (c(u) - 2)$$

$$o_{24} + 4o_{65} + 4o_{62} + o_{59} + 6o_{56} + o_{51} + 2o_{45} + 2o_{39} = \sum_{u,v,t\colon P_{9}(x,u,v,t)} (c(v) - 2) + (c(t) - 2)$$

$$4o_{23} + o_{55} + o_{42} + 2o_{33} = \sum_{u,v,t\colon P_{7}(x,u,v,t)} c(x) - 3$$

$$3o_{22} + 2o_{54} + o_{40} + o_{39} + o_{32} + 2o_{31} = \sum_{u,v,t\colon P_{6}(x,u,v,t)} c(u) - 3$$

$$o_{21} + 3o_{55} + 3o_{50} + 2o_{42} + 2o_{38} + 2o_{33} = \sum_{u,v,t\colon P_7(x,u,v,t)} (c(u) - 1) + (c(v) - 1) + (c(t) - 1)$$

$$o_{20} + 2o_{54} + 2o_{49} + o_{40} + o_{37} + o_{32} = \sum_{u,v,t\colon P_6(x,u,v,t)} c(x) - 1$$

$$o_{19} + 4o_{54} + 4o_{49} + o_{40} + 2o_{39} + o_{37} + 2o_{35} + 2o_{31} = \sum_{u,v,t\colon P_6(x,u,v,t)} (c(v) - 1) + (c(t) - 1)$$

$$2o_{18} + o_{59} + o_{51} + 2o_{46} + 2o_{45} + 2o_{36} + 2o_{27} + o_{24} = \sum_{u,v,t\colon P_4(x,u,v,t)} c(v) - 2$$

$$2o_{17} + o_{60} + o_{53} + o_{51} + o_{48} + o_{37} + 2o_{34} + 2o_{30} = \sum_{u,v,t\colon P_5(x,u,v,t)} c(u) - 1$$

$$o_{16} + o_{59} + 2o_{52} + o_{51} + 2o_{46} + 2o_{36} + 2o_{34} + o_{29} = \sum_{u,v,t\colon P_4(x,u,v,t)} c(x) - 1$$

$$o_{15} + o_{59} + 2o_{52} + o_{51} + 2o_{45} + 2o_{35} + 2o_{34} + 2o_{27} = \sum_{u,v,t\colon P_4(x,u,v,t)} c(t) - 1$$

D. Equations for edge-orbit counts in 5-graphlets

Conditions, P_i, define the order of nodes and put edge (x, y) in orbit E_i; e.g., in P_{13} edge (x, y) is in orbit E_{13}.

$$\begin{aligned}
P_{11}(x,y,a,b) &= a < b \wedge G[\{x,y,a,b\}] \cong G_8\\
P_{10}(x,y,a,b) &= a < b \wedge (a,b) \notin E \wedge G[\{x,y,a,b\}] \cong G_7\\
P_{9a}(x,y,a,b) &= a \in N(x) \wedge b \in N(y) \wedge (a,b) \in E \wedge (x,b) \in E \wedge G[\{x,y,a,b\}] \cong G_7\\
P_{9b}(x,y,a,b) &= a \in N(x) \wedge b \in N(y) \wedge (a,b) \in E \wedge (y,a) \in E \wedge G[\{x,y,a,b\}] \cong G_7\\
P_7(x,y,a,b) &= a \in N(x) \cap N(y) \wedge b \in N(a) \wedge G[\{x,y,a,b\}] \cong G_6\\
P_{6a}(x,y,a,b) &= a < b \wedge (a,b) \in E \wedge a,b \in N(y) \wedge a,b \notin N(x) \wedge G[\{x,y,a,b\}] \cong G_6\\
P_{6b}(x,y,a,b) &= a < b \wedge (a,b) \in E \wedge a,b \in N(x) \wedge a,b \notin N(y) \wedge G[\{x,y,a,b\}] \cong G_6\\
P_5(x,y,a,b) &= a \in N(x) \wedge b \in N(y) \wedge G[\{x,y,a,b\}] \cong G_5\\
P_{4a}(x,y,a,b) &= a < b \wedge a,b \in N(y) \wedge G[\{x,y,a,b\}] \cong G_4\\
P_{4b}(x,y,a,b) &= a < b \wedge a,b \in N(x) \wedge G[\{x,y,a,b\}] \cong G_4\\
P_3(x,y,a,b) &= a \in N(x) \wedge b \in N(y) \wedge G[\{x,y,a,b\}] \cong G_3\\
P_{2a}(x,y,a,b) &= (a,b) \in E \wedge a \in N(y) \wedge G[\{x,y,a,b\}] \cong G_3\\
P_{2b}(x,y,a,b) &= (a,b) \in E \wedge a \in N(x) \wedge G[\{x,y,a,b\}] \cong G_3
\end{aligned}$$

Equations:

$$2e_{66} + 6e_{67} = \sum_{a,b\colon P_{11}(x,y,a,b)} (c(x,y,a) + c(x,y,b) - 2)$$

$$e_{65} + 6e_{67} = \sum_{a,b\colon P_{11}(x,y,a,b)} (c(x,a,b) + c(y,a,b) - 2)$$

$$e_{64} + 2e_{66} = \sum_{a,b\colon P_{10}(x,y,a,b)} (c(x,a,b) + c(y,a,b) - 2)$$

$$2e_{63}+2e_{65}=\sum_{a,b\colon P_{9a}(x,y,a,b)}(c(y,a,b)-1)+\sum_{a,b\colon P_{9b}(x,y,a,b)}(c(x,a,b)-1)$$

$$e_{62}+2e_{66}+3e_{67}=\sum_{a,b\colon P_{11}(x,y,a,b)}(c(x,y)-2)$$

$$e_{61}+2e_{65}+4e_{66}+12e_{67}=\sum_{a,b\colon P_{11}(x,y,a,b)}(c(x,a)+c(x,b)+c(y,a)+c(y,b)-8)$$

$$e_{60}+e_{65}+3e_{67}=\sum_{a,b\colon P_{11}(x,y,a,b)}(c(a,b)-2)$$

$$2e_{59}+2e_{65}=\sum_{a,b\colon P_{9a}(x,y,a,b)}c(x,a,b)+\sum_{a,b\colon P_{9b}(x,y,a,b)}c(y,a,b)$$

$$e_{58}+e_{64}+e_{66}=\sum_{a,b\colon P_{10}(x,y,a,b)}(c(a,b)-2)$$

$$e_{57}+2e_{63}+2e_{64}+2e_{65}=\sum_{a,b\colon P_{9a}(x,y,a,b)}(c(y,a)-2)+\sum_{a,b\colon P_{9b}(x,y,a,b)}(c(x,b)-2)$$

$$2e_{56}+2e_{63}=\sum_{a,b\colon P_{5}(x,y,a,b)}(c(x,a,b)+c(y,a,b))$$

$$e_{55}+4e_{62}+2e_{64}+4e_{66}=\sum_{a,b\colon P_{10}(x,y,a,b)}(c(x,a)+c(x,b)+c(y,a)+c(y,b)-4)$$

$$2e_{54}+e_{61}+2e_{63}+2e_{65}=\sum_{a,b\colon P_{9a}(x,y,a,b)}(c(y,b)-1)+\sum_{a,b\colon P_{9b}(x,y,a,b)}(c(x,a)-1)$$

$$e_{53}+2e_{59}+2e_{64}+2e_{65}=\sum_{a,b\colon P_{9a}(x,y,a,b)}(c(x,a)-1)+\sum_{a,b\colon P_{9b}(x,y,a,b)}(c(y,b)-1)$$

$$e_{52}+2e_{59}+2e_{63}+2e_{65}=\sum_{a,b\colon P_{9a}(x,y,a,b)\vee P_{9b}(x,y,a,b)}(c(a,b)-1)$$

$$e_{51}+e_{61}+2e_{62}+e_{65}+4e_{66}+6e_{67}=\sum_{a,b\colon P_{11}(x,y,a,b)}(c(x)+c(y)-6)$$

$$e_{50}+2e_{60}+e_{61}+2e_{65}+2e_{66}+6e_{67}=\sum_{a,b\colon P_{11}(x,y,a,b)}(c(a)+c(b)-6)$$

$$3e_{49}+e_{59}=\sum_{a,b\colon P_{6a}(x,y,a,b)}c(y,a,b)+\sum_{a,b\colon P_{6b}(x,y,a,b)}c(x,a,b)$$

$$3e_{48}+2e_{62}+e_{66}=\sum_{a,b\colon P_{10}(x,y,a,b)}(c(x,y)-2)$$

$$2e_{47}+2e_{59}+e_{61}+2e_{65}=\sum_{a,b\colon P_{9a}(x,y,a,b)}(c(x,b)-2)+\sum_{a,b\colon P_{9b}(x,y,a,b)}(c(y,a)-2)$$

$$e_{46}+e_{57}+e_{63}=\sum_{a,b\colon P_{5}(x,y,a,b)}c(x,y)$$

$$e_{45}+e_{52}+4e_{58}+4e_{60}=\sum_{a,b\colon P_{7}(x,y,a,b)}(c(x,b)+c(y,b)-2)$$

$$e_{44}+2e_{56}+e_{57}+2e_{63}=\sum_{a,b\colon P_{5}(x,y,a,b)}(c(x,a)+c(y,b))$$

$$e_{43}+2e_{56}+e_{63}=\sum_{a,b\colon P_{5}(x,y,a,b)}c(a,b)$$

$$2e_{42} + 2e_{56} + e_{57} + 2e_{63} = \sum_{a,b\colon P_5(x,y,a,b)} (c(x,b) + c(y,a) - 4)$$

$$e_{41} + e_{55} + 2e_{58} + 2e_{62} + 2e_{64} + 2e_{66} = \sum_{a,b\colon P_{10}(x,y,a,b)} (c(a) + c(b) - 4)$$

$$\begin{aligned} e_{40} + 2e_{54} + e_{55} + e_{57} + e_{61} + 2e_{63} + 2e_{64} + 2e_{65} &= \\ &\sum_{a,b\colon P_{9a}(x,y,a,b)} (c(y) - 2) + \sum_{a,b\colon P_{9b}(x,y,a,b)} (c(x) - 2) \end{aligned}$$

$$\begin{aligned} e_{39} + e_{52} + e_{53} + e_{57} + 2e_{59} + 2e_{63} + 2e_{64} + 2e_{65} &= \\ &\sum_{a,b\colon P_{9a}(x,y,a,b)} (c(a) - 2) + \sum_{a,b\colon P_{9b}(x,y,a,b)} (c(b) - 2) \end{aligned}$$

$$e_{38} + 3e_{49} + e_{56} + e_{59} = \sum_{a,b\colon P_{6a}(x,y,a,b) \vee P_{6b}(x,y,a,b)} (c(a,b) - 1)$$

$$e_{37} + e_{53} + e_{59} = \sum_{a,b\colon P_{6a}(x,y,a,b) \vee P_{6b}(x,y,a,b)} c(x,y)$$

$$2e_{36} + e_{52} + 2e_{60} = \sum_{a,b\colon P_7(x,y,a,b)} c(a,b)$$

$$e_{35} + 6e_{48} + e_{55} + 4e_{62} + e_{64} + 2e_{66} = \sum_{a,b\colon P_{10}(x,y,a,b)} (c(x) + c(y) - 6)$$

$$\begin{aligned} e_{34} + 2e_{47} + e_{53} + e_{55} + 2e_{59} + e_{61} + 2e_{64} + 2e_{65} &= \\ &\sum_{a,b\colon P_{9a}(x,y,a,b)} (c(x) - 3) + \sum_{a,b\colon P_{9b}(x,y,a,b)} (c(y) - 3) \end{aligned}$$

$$\begin{aligned} e_{33} + 2e_{47} + e_{52} + 2e_{54} + 2e_{59} + e_{61} + 2e_{63} + 2e_{65} &= \\ &\sum_{a,b\colon P_{9a}(x,y,a,b)} (c(b) - 3) + \sum_{a,b\colon P_{9b}(x,y,a,b)} (c(a) - 3) \end{aligned}$$

$$\begin{aligned} 2e_{32} + 6e_{49} + e_{53} + 2e_{59} = &\sum_{a,b\colon P_{6a}(x,y,a,b)} (c(y,a) + c(y,b) - 2) + \\ &\sum_{a,b\colon P_{6b}(x,y,a,b)} (c(x,a) + c(x,b) - 2) \end{aligned}$$

$$e_{31} + 2e_{42} + e_{44} + 2e_{46} + 2e_{56} + 2e_{57} + 2e_{63} = \sum_{a,b\colon P_5(x,y,a,b)} (c(x) + c(y) - 4)$$

$$e_{30} + 2e_{42} + 2e_{43} + e_{44} + 4e_{56} + e_{57} + 2e_{63} = \sum_{a,b\colon P_5(x,y,a,b)} (c(a) + c(b) - 4)$$

$$2e_{29} + 2e_{38} + e_{45} + e_{52} = \sum_{a,b\colon P_{2a}(x,y,a,b)} (c(y,b) - 1) + \sum_{a,b\colon P_{2b}(x,y,a,b)} (c(x,b) - 1)$$

$$2e_{28} + 2e_{43} + e_{45} + e_{52} = \sum_{a,b\colon P_{2a}(x,y,a,b)} c(x,b) + \sum_{a,b\colon P_{2b}(x,y,a,b)} c(y,b)$$

$$e_{27} + e_{34} + e_{47} = \sum_{a,b\colon P_{4a}(x,y,a,b) \vee P_{4b}(x,y,a,b)} c(x,y)$$

$$2e_{26} + e_{33} + 2e_{36} + e_{50} + e_{52} + 2e_{60} = \sum_{a,b\colon P_7(x,y,a,b)} (c(a) - 3)$$

$$e_{25} + 2e_{32} + e_{37} + 3e_{49} + e_{53} + e_{59} = \sum_{a,b\colon P_{6a}(x,y,a,b)} (c(y) - 3) + \sum_{a,b\colon P_{6b}(x,y,a,b)} (c(x) - 3)$$

$$e_{24} + e_{39} + e_{45} + e_{52} = \sum_{a,b\colon P_{2a}(x,y,a,b) \vee P_{2b}(x,y,a,b)} c(x,y)$$

$$e_{23} + 2e_{36} + e_{45} + e_{52} + 2e_{58} + 2e_{60} = \sum_{a,b\colon P_{7}(x,y,a,b)} (c(b) - 1)$$

$$e_{22} + e_{37} + e_{44} + e_{53} + e_{56} + e_{59} = \sum_{a,b\colon P_{6a}(x,y,a,b)} (c(x) - 1) + \sum_{a,b\colon P_{6b}(x,y,a,b)} (c(y) - 1)$$

$$2e_{21} + 2e_{38} + 2e_{43} + e_{52} = \sum_{a,b\colon P_{2a}(x,y,a,b) \vee P_{2b}(x,y,a,b)} c(a,b)$$

$$e_{20} + e_{40} + e_{54} = \sum_{a,b\colon P_{3}(x,y,a,b)} c(x,y)$$

$$e_{19} + e_{33} + 2e_{41} + e_{45} + 2e_{50} + e_{52} + 4e_{58} + 4e_{60} = \sum_{a,b\colon P_{7}(x,y,a,b)} (c(x) + c(y) - 4)$$

$$e_{18} + 2e_{32} + 2e_{38} + e_{44} + 6e_{49} + e_{53} + 2e_{56} + 2e_{59} = \sum_{a,b\colon P_{6a}(x,y,a,b) \vee P_{6b}(x,y,a,b)} (c(a) + c(b) - 4)$$

$$3e_{17} + 2e_{25} + e_{27} + e_{32} + e_{34} + e_{47} = \sum_{a,b\colon P_{4a}(x,y,a,b)} (c(y) - 3) + \sum_{a,b\colon P_{4b}(x,y,a,b)} (c(x) - 3)$$

$$2e_{16} + 2e_{20} + 2e_{22} + e_{31} + 2e_{40} + e_{44} + 2e_{54} = \sum_{a,b\colon P_{3}(x,y,a,b)} (c(x) + c(y) - 4)$$

$$e_{15} + 2e_{25} + 2e_{29} + e_{31} + 2e_{32} + e_{34} + 2e_{42} + 2e_{47} = \sum_{a,b\colon P_{4a}(x,y,a,b) \vee P_{4b}(x,y,a,b)} (c(a) + c(b) - 2)$$

$$2e_{14} + e_{18} + 2e_{21} + e_{30} + 2e_{38} + e_{39} + 2e_{43} + e_{52} = \sum_{a,b\colon P_{2a}(x,y,a,b) \vee P_{2b}(x,y,a,b)} (c(a) - 2)$$

$$e_{13} + 2e_{22} + 2e_{28} + e_{31} + e_{40} + 2e_{44} + 2e_{54} = \sum_{a,b\colon P_{3}(x,y,a,b)} (c(a) + c(b) - 2)$$

$$e_{12} + 2e_{21} + 2e_{28} + 2e_{29} + 2e_{38} + 2e_{43} + e_{45} + e_{52} = \sum_{a,b\colon P_{2a}(x,y,a,b) \vee P_{2b}(x,y,a,b)} (c(b) - 1)$$

Affiliation:

Tomaž Hočevar, Janez Demšar
Bioinformatics Laboratory
Faculty of Computer and Information Science, University of Ljubljana
Večna pot 113, 1000 Ljubljana, Slovenia
E-mail: tomaz.hocevar@fri.uni-lj.si, janez.demsar@fri.uni-lj.si
URL: http://www.fri.uni-lj.si/en/tomaz-hocevar,
http://www.fri.uni-lj.si/en/janez-demsar

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
July 2016, Volume 71, Issue 10 *Submitted:* 2014-06-23
doi:10.18637/jss.v071.i10 *Accepted:* 2015-06-11