Testing Cluster Properties of Signed Graphs

Florian Adriaens, Simon Apers

Testing Cluster Properties of Signed Graphs

Florian Adriaens*
florian.adriaens@helsinki.fi
HIIT, University of Helsinki
Helsinki, Finland

Simon Apers
smgapers@gmail.com
Université de Paris, CNRS, IRIF
Paris, France

ABSTRACT

This work initiates the study of property testing in signed graphs, where every edge has either a positive or a negative sign. We show that there exist sublinear query and time algorithms for testing three key properties of signed graphs: *balance* (or *2-clusterability*), *clusterability* and *signed triangle freeness*. We consider both the dense graph model, where one queries the adjacency matrix entries of a signed graph, and the bounded-degree model, where one queries for the neighbors of a node and the sign of the connecting edge. Our algorithms use a variety of tools from unsigned graph property testing, as well as reductions from one setting to the other. Our main technical contribution is a sublinear algorithm for testing clusterability in the bounded-degree model. This contrasts with the property of k-clusterability in unsigned graphs, which is not testable with a sublinear number of queries in the bounded-degree model. We experimentally evaluate the complexity and usefulness of several of our testers on real-life and synthetic datasets.

CCS CONCEPTS

• **Theory of computation** → Design and analysis of algorithms; • **Mathematics of computing** → Discrete mathematics.

KEYWORDS

signed graphs, property testing, random walks, clustering, social networks

ACM Reference Format:
Florian Adriaens and Simon Apers. 2023. Testing Cluster Properties of Signed Graphs. In *Proceedings of the ACM Web Conference 2023 (WWW '23), April 30–May 04, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3543507.3583213

1 INTRODUCTION

A *signed graph* is a graph where every edge either has a positive or a negative label. Formally it is denoted as $G = (V, E, \sigma)$ with node set $V = [N]$, edge set $E \subseteq V \times V$ and edge labelling $\sigma : E \to \{+, -\}$. Such graphs model a variety of different scientific phenomena. The widely studied correlation clustering problem [7, 14] was orginally motivated by a document classification problem, where one has knowledge of pairwise similarities between documents, and the goal is to cluster the documents into (an unspecified number of) groups such that within a group the documents are similar to each other, while across groups they are less similar. Several authors [9, 37] have focused on the related problem of finding large polarized communities in signed graphs. The survey of [36] lists several other important mining tasks in signed social media platforms.

A second example are web applications such as online social networks. Interactions between individuals on these platforms can often be categorized into binary categories: trust versus distrust, friendly or antagonistic, etc. An important aspect in the edge formation of social networks is the sign of triangles. According to structural balance theory from social psychology [10], triangles with either one or three positive edges are more plausible, and this prevalence has been observed in real-life social networks [30, 36]. Many methods and algorithms for link and sign prediction try to capitalize on this [11, 30, 32]. For a given sign configuration of a triangle (e.g, a $\{+, +, -\}$ triangle) we say that a signed graph is *signed triangle free* if such a triangle is not present in the graph. The presence of signed triangles is relevant in structural balance theory [10], and $\{+, +, -\}$ triangle packings have been used as lower bounds in designing approximation algorithms for correlation clustering problems [1, 7, 38]. A first natural question we ask is if we can quickly decide whether a signed graph is *signed triangle free* or "far" from it. This turns out to be very similar to the unsigned case, so we shift our focus to *cluster* properties inherently specific to signed graphs.

Since signed graphs generalize unsigned graphs, they can have different properties. One important example is the property of *clusterability* or *weak balance*, this was first introduced in [13] and it is the subject of the correlation clustering problem [7]. A signed graph is clusterable if there exists a partitioning of the nodes into an a priori unknown number of components such that (i) every positive edge connects two nodes in the same component, and (ii) every negative edge connects two nodes in different components. An equivalent characterization, in terms of forbidden subgraphs, is that the signed graph contains no cycles with exactly one negative edge [13, 14]. In Section 3.2 we utilize this forbidden subgraph characterization to design a clusterability tester in the bounded-degree model with query and time complexity $\tilde{O}(\sqrt{N}/\mathrm{poly}(\epsilon))$. Clusterability does not appear to have a meaningful interpretation in the case of unsigned graphs. For example, if one views an unsigned graph as a signed graph with the restriction that all the edges have the same sign, whether positive or negative, then clearly any such graph is clusterable since it contains no cycles with exactly one negative edge.

Other signed graph properties are closer related to unsigned graph properties. For example, the property of *balance* or *2-clusterability*) in signed graphs [23, 25] in fact generalizes that of bipartiteness in unsigned graphs. A signed graph is balanced if it is clusterable into exactly two components, with only positive edges

*Part of this work was done when the author was at KTH, Stockholm, Sweden.

WWW '23, April 30–May 04, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04.
https://doi.org/10.1145/3543507.3583213

inside the components and negative edges between the components. It follows that a signed graph with only negative edges is balanced if and only if the underlying unsigned graph is bipartite. There is also a distance-preserving reduction in the opposite direction, transforming a signed graph to an unsigned graph by replacing each positive edge by a path of two negative edges, and afterwards omitting all the signs of the edges [39]. We use this reduction to show that in the bounded-degree model we can reduce the problem of testing balance to that of testing bipartiteness.[1]

1.1 Graph Property Testing: What and Why?

Graph property testing was formally introduced in the seminal work of Goldreich, Goldwasser and Ron [20]. As input we are given query access to an (unsigned) graph $G = (V, E)$ with node set $V = [N]$ and edge set $E \subseteq V \times V$. We would like to decide whether the graph obeys a certain property $\mathcal{P}$, or whether it is "far" from any graph having that property. This is a relaxed setting as compared to that of deciding $\mathcal{P}$ and it often allows for algorithms that have sublinear query and/or time complexity. Such sublinear algorithms have been proposed for a wide range of graph properties such as bipartiteness [20, 21], k-colorability [3, 20], cycle-freeness [12, 22] and more generally monotone graph properties [4] and minor-closed properties [26, 27]. The precise definition of "far" however depends on the type of query access that we have to the graph:

(1) In the *dense graph model* [19] we are able to query the adjacency matrix entries. A query takes the form $(v, w) \in [N] \times [N]$ and the reply is 1 if there is an edge between v and w, otherwise it is 0. Two graphs $G = (V, E)$ and $G' = (V, E')$ are said to be ϵ-far from each other if they differ in at least an ϵ-fraction of the adjacency matrix entries. Equivalently, at least ϵN^2 edges have to be added or removed to turn G into G'.
(2) In the *bounded-degree graph model* we are given an upper bound d on the degrees of the graph, and we are given access to the adjacency list of G. A query takes the form (v, i), with $v \in [N]$ and $i \in [d]$. If the degree of v is at least i, then the query is answered with the i^{th} neighbor u of node v (in arbitrary order). If v has degree smaller than i, an error symbol is returned. Two graphs $G = (V, E)$ and $G' = (V, E')$ are ϵ-far from each other if at least ϵNd edges have to be modified (added or removed) to turn G into G'.

Using these definitions, a graph is *ϵ-far from having property $\mathcal{P}$* if it is ϵ-far from any graph having property $\mathcal{P}$. A property testing algorithm for $\mathcal{P}$ is a randomized algorithm[2] that, given query access to G and an error parameter ϵ, must behave as follows [20]:

(1) if G has property $\mathcal{P}$ then the algorithm should accept with probability at least 2/3.
(2) if G is ϵ-far from having property $\mathcal{P}$, then the algorithm should reject with probability at least 2/3.

If G satisfies neither condition than the algorithm can behave arbitrarily. This is the main reason why testing algorithms are often far more efficient than algorithms for effectively deciding whether G has property $\mathcal{P}$ or not. If a property tester always accepts graphs having property $\mathcal{P}$ (i.e., it never falsely rejects), then it is called a *one-sided* tester for $\mathcal{P}$. Otherwise it is called *two-sided*. If $\mathcal{P}$ can be tested by a tester with query complexity independent of N (only dependent on ϵ), then $\mathcal{P}$ is said to be *easily testable*.

There are several potential benefits of using a property testing algorithm (see also [20]). First, they can be used before running an algorithm that decides $\mathcal{P}$ exactly. If a graph is far from satisfying $\mathcal{P}$, we obtain a proof of this with good probability, and save the time of running the exact algorithm. If the tester accepts, we may still run the exact algorithm without much extra time. Second, in case one is promised that input graphs are either good or very bad, then there is no need for the exact algorithm at all. Third, one might not have the time to run the exact algorithm before making design decisions. This is helpful in scenarios where a good design consists of an input satisfying $\mathcal{P}$. A tester will accept good designs and reject designs that need a lot of cost (edge modifications) to become a good design. The tester may still accept designs that are not good, but then at least we know there is a little cost needed to make it into a good design. Lastly, sometimes the ideas in the design and analysis of testing algorithms can be used to design distance approximation algorithms. We refer the interested reader to the survey by Goldreich [18] for more connections to sublinear time approximation, streaming algorithms and computational learning theory. We also mention the important result of [16] that in dense graphs if a property is easily testable, then it is possible to estimate how far (how many edge modifications are required) to an input graph is to satisfying the property, within additive error δn^2, in time that depends only on δ.

Signed Graph Testing Framework. The definitions of distance and query access for unsigned graphs are easily extended to signed graphs:

(1) In the *dense signed graph model* adjacency matrix queries are now answered by an element from $\{0, -, +\}$. A signed graph is ϵ-far from property $\mathcal{P}$ if at least ϵN^2 edge modifications (addition, removal or sign switch) have to be made to obtain a graph that satisfies $\mathcal{P}$.
(2) In the *bounded-degree signed graph model* a query $(v, i) \in [N] \times [d]$ is now answered either by the i-th neighbor w of v and the sign $\sigma(v, w)$ of the corresponding edge, or by an error symbol if v has less than i neighbors. A signed graph is ϵ-far from $\mathcal{P}$ if at least ϵnd edge modifications (addition, removal or sign switch) have to be made to obtain a graph that satisfies $\mathcal{P}$.

In both models the edge modifications consist of edge additions, removals, as well as sign switches. Since the properties discussed in this paper (signed triangle freeness, balance and clusterability) are all monotone, we restrict our attention to edge removals.

1.2 Results and Techniques

Table 1 summarizes our results in terms of query complexity. The $\tilde{O}(\cdot)$-notation hides polylogarithmic factors in its argument and in N, in the bounded-degree model it also hides a polynomial dependence on the degree bound d. The time complexity for testing balance and clusterability in the bounded-degree model is bounded by the query complexity $\tilde{O}(\sqrt{N}/\text{poly}(\epsilon))$. In all other cases the

[1] This reduction does not work well in the dense graph model, since the transformed graph will be typically sparse.
[2] The accept and reject probabilities are chosen arbitrarily and can be amplified by standard techniques.

Table 1: Query complexity of the three testers in the two models. All testers are one-sided except for the clusterability tester in the dense model. The function $\text{tower}(\log(1/\epsilon))$ **denotes a power tower of 2's of height** $O(\log(1/\epsilon))$**.**

	Dense signed graph model	Bounded-degree signed graph model
Signed triangle freeness	$\tilde{O}(\text{tower}(\log(1/\epsilon)))$ [17]	$\tilde{O}(1/\epsilon)$ [18]
Balance	$\tilde{O}(1/\epsilon^2)$ [35]	$\tilde{O}(\sqrt{N}/\text{poly}(\epsilon))$ **[this work]**
Clusterability	$\tilde{O}(1/\epsilon^7)$ **[this work]**	$\tilde{O}(\sqrt{N}/\text{poly}(\epsilon))$ **[this work]**

time complexity is at most exponential in the query complexity. In the rest of the section we give a sketch of our techniques.

Dense signed graph model. The property of *signed triangle freeness* can be efficiently tested by interpreting the signed graph as an edge-colored graph. We can then use Fox's edge-colored triangle removal lemma [17], similar to the case of triangle freeness in unsigned graphs. For the property of *balance* or 2-clusterability we can use a reduction to a Constraint Satisfaction Problem (CSP): every node corresponds to a Boolean variable (which indicates its cluster), a positive edge imposes an equality constraint between its endpoints and a negative edge imposes an inequality constraint. We can then test balance by using a property testing algorithm for CSPs [2, 6, 35]. The property of *clusterability* can also be cast as a CSP in which the node variables now take arbitrary integer values in $[N]$ (indicating their cluster). However, the aforementioned CSP testers [2, 6, 35] are not efficient in such a regime. We circumvent this problem by proving that a signed graph that is clusterable is necessarily $\epsilon/10$-close to being clusterable into $O(1/\epsilon)$ clusters. Using this we reduce the problem of testing clusterability to that of distinguishing graphs that are $\epsilon/10$-close to being $O(1/\epsilon)$-clusterable from those that are ϵ-far from being $O(1/\epsilon)$-clusterable. This problem corresponds to *tolerantly* testing a CSP where the variables now take values in $[O(1/\epsilon)]$, and this can be done efficiently using an algorithm by Andersson and Engebretsen [6].

Bounded-degree signed graph model. Testing signed triangle freeness is trivial in this model, similar to the unsigned case [18]. Testing balance requires more care. While we can again cast the problem as a CSP, we are not aware of any appropriate property testing algorithms for CSPs in the bounded-degree model. Rather we reduce the problem of testing balance for signed graphs to that of testing bipartiteness for unsigned graphs, for which we can use the algorithm of Goldreich and Ron [21]. The reduction is based on a transformation described by Zaslavsky [39], which maps balanced (resp. unbalanced) signed graphs to bipartite (resp. nonbipartite) unsigned graphs. The resulting algorithm's query complexity has an optimal $\sqrt{N}$-dependence, which follows from the $\Omega(\sqrt{N})$ lower bound for testing bipartiteness.

Finally, and this is our main technical contribution, we describe a property testing algorithm for clusterability. While we can again reduce the problem to testing $O(1/\epsilon)$-clusterability, similar to the dense case, the problem is that k-colorability (which is a special case of k-clusterability) is not testable in the bounded-degree model [8]. Rather, we base our algorithm on the forbidden subgraph characterization by Davis [13], which states that a signed graph is clusterable if and only if it has no cycles with exactly one negative edge. We then use random walks to find such a cycle: first we pick a random initial node and perform a large number of random walks *on the positive edges* of G, then we check for the existence of a negative edge between any pair of nodes that were visited by a random walk. Such a negative edge necessarily yields a bad cycle. The correctness of this algorithm is straightforward to prove when the positive edges in G induce an expander. For the general case we build on the (unsigned) graph decomposition results of [21].

2 DENSE SIGNED GRAPH MODEL

Testing signed triangle freeness in the dense model can be done by using Fox's *edge-colored* triangle removal lemma [17], similarly like testing triangle freeness in unsigned graphs is a direct application of the triangle removal lemma [17, 34]. More details and the proof of Theorem 1 can be found in Appendix A.1.

Theorem 1. *There exists a one-sided tester for signed triangle freeness in the dense signed graph model with query complexity* $\tilde{O}(\text{tower}(\log(1/\epsilon)))$.

2.1 Balance

We cast balance or 2-clusterability of a signed graph $G = (V, E, \sigma)$ as a satisfiability problem. Associate with each node v a variable $x_v \in \{0, 1\}$. With every edge $(u, v) \in E$ we associate a constraint on x_u and x_v: if $\sigma(e) = +$ (positive edge) the constraint is satisfied iff $x_u = x_v$; if $\sigma(e) = -$ (negative edge) then the constraint is satisfied iff $x_u \neq x_v$. The graph G will be balanced iff there exists an assignment of x_v's such that all constraints are satisfied. Even more, if G is ϵ-far from being balanced then we have to remove ϵn^2 constraints from the satisfiability problem for it to be satisfiable. As a consequence, the problem reduces to testing whether the satisfiability problem is in fact satisfiable. For this we can use the work by Sohler [35] which describes a one-sided tester with query complexity $\tilde{O}(1/\epsilon^2)$. The algorithm is very simple: sample $\tilde{O}(1/\epsilon)$ variables, query the whole induced subgraph and accept if and only if the induced set of constraints on those variables has a satisfying assignment. Applying this algorithm to the problem of testing balance gives the following algorithm: sample $\tilde{O}(1/\epsilon)$ nodes, query the entire induced subgraph, and accept if and only if the induced subgraph is balanced. From [35, Theorem 1] it then follows that this describes a one-sided property tester for balance. This proves Theorem 2.

Theorem 2. *There exists a one-sided tester for balance in the dense signed graph model with query complexity* $\tilde{O}(1/\epsilon^2)$.

Note that testing k-clusterability can be reduced to satisfiability in a similar manner, except that now the variables $x_v \in \{0, 1, \ldots, k-1\}$. For constant k the result of Sohler [35, Theorem 1] thus implies a one-sided tester for k-clusterability with query complexity $\tilde{O}(1/\epsilon^2)$.

2.2 Clusterability

A relaxation of k-clusterability for signed graphs is the notion of *weak balance* or *clusterability* [7, 13]. A signed graph is clusterable if it is k-clusterable for some (a priori unknown) $k \in [N]$. Since there can be at most N clusters, we could test clusterability by testing N-clusterability. The satisfiability reduction from last section however fails in such case, because typical satisfiability testers have a bad dependence on the domain size of the variables.

Rather, we argue that testing clusterability can be reduced to *tolerantly* testing k-clusterability for $k \in O(1/\epsilon)$. A tolerant tester [33] is required to accept inputs that are ϵ_1-close to some property $\mathcal{P}$, while rejecting inputs that are ϵ_2-far from $\mathcal{P}$, for some parameters $\epsilon_1 < \epsilon_2$. Tolerant testing is closely related to approximating the distance from an object to a property (see [33]). We use the following lemma.

Lemma 3. *If a signed graph is clusterable then it is 4ϵ-close to being clusterable into at most $1/\epsilon$ clusters.*

Proof. Let the partition $V = P_1 \cup P_2 \cup \cdots \cup P_r$ denote a valid clustering of the graph. We define a new partition by merging different components: keep all components P_i of size $|P_i| \geq \epsilon N$, and merge the remaining components into components of size between ϵN and $2\epsilon N$ (which is always possible). This yields a new partition with at most $1/\epsilon$ components. Between the components there are only negative edges, and there are at most $(2\epsilon N)^2/\epsilon = 4\epsilon N^2$ edges within the components. Hence if we remove all the edges within the new components, then we obtain a new graph for which the new partition describes a clustering with at most $1/\epsilon$ clusters, and which is (4ϵ)-far from the original graph. □

Now if a signed graph is ϵ-far from being clusterable, then clearly it is also ϵ-far from being say $(8/\epsilon)$-clusterable. On the other hand, by this lemma, a graph that is $\epsilon/4$-close to being clusterable will be $(\epsilon/4 + \epsilon/2) = 3\epsilon/4$-close to a graph that is $(8/\epsilon)$-clusterable. Hence we can use a *tolerant tester* for $O(1/\epsilon)$-clusterability to tolerantly test clusterability. Equivalently, we can use an additive estimate (with error $\pm\epsilon N^2$) on the number of edges that need to be removed in order to make a signed graph k-clusterable, for $k \in O(1/\epsilon)$. Now we are in better shape to cast the problem as a satisfiability problem, similar to last section. In Appendix B.1 we detail how to use the algorithm of Andersson and Engebretsen [6] to tolerantly test for $O(1/\epsilon)$-clusterability using $\tilde{O}(1/\epsilon^7)$ queries. This yields the following theorem.

Theorem 4. *There exists a two-sided tolerant tester for clusterability in the dense signed graph model with query complexity $\tilde{O}(1/\epsilon^7)$.*

Moreover, since the tester is tolerant it also estimates (within additive error ϵN^2) the minimum number of of edge deletions such that the remaining graph is clusterable by using $\tilde{O}(1/\epsilon^7)$ queries.

3 TESTING IN THE BOUNDED-DEGREE SIGNED GRAPH MODEL

Testing signed triangle freeness is similar to the unsigned case [18]. The algorithm picks $\tilde{O}(1/\epsilon)$ nodes and rejects the input graph if any of them is part of a signed triangle. The proof of Theorem 5 can be found in Appendix A.2.

Theorem 5. *There exists a one-sided tester for signed triangle freeness in the bounded-degree model with query complexity $\tilde{O}(1/\epsilon)$.*

3.1 Balance

Our algorithm for testing balance of bounded-degree signed graphs reduces the problem to testing bipartiteness in a related unsigned graph[3]. Consider the following mapping (Figure 1) from a signed graph G to an unsigned graph G': (i) for every positive edge (u, v) create a new node $w^{(u,v)}$ and replace the edge (u, v) by two unsigned edges $(u, w^{(u,v)})$ and $(w^{(u,v)}, v)$, and (ii) replace each of the remaining negative edges by an unsigned edge. The unsigned graph G' has an odd cycle if and only if G has a cycle with an odd number of negative edges. As a consequence, G' will be bipartite if and only if G was balanced.

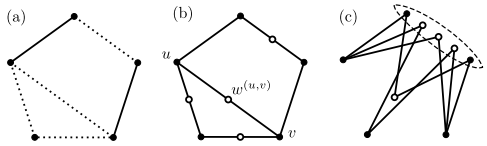


Figure 1: (a) Balanced signed graph G. Dotted lines are positive edges, solid lines are negative edges. (b) Bipartite unsigned graph G' after the mapping. (c) Bipartition of G'.

The *signed frustration index* of a signed graph is the minimum number of required edge deletions to make the resulting graph balanced. The *unsigned frustration index* of an unsigned graph is the minimum number of required edge deletions to make it bipartite. In [39, Proposition 2.2] a stronger result was proven, namely that the (signed) frustration index of G is equal to the (unsigned) frustration index of G'. This implies the following lemma.

Lemma 6. *If G is ϵ-far from balanced then G' is $\epsilon/(d+1)$-far from bipartite.*

Proof. For the second fact, let G be ϵ-far from being balanced, so that it has signed frustration index $k \geq \epsilon dN$. The unsigned graph G' then has unsigned frustration index $k \geq \epsilon dN$. Now if G has m^+ positive edges, then G' has exactly $N + m^+ \leq (d+1)N$ vertices while keeping the same degree bound d. As a consequence, we can bound its frustration index $k \geq \frac{\epsilon}{d+1}(d+1)dN \geq \frac{\epsilon}{d+1}d(N+m^+)$, so that G' is indeed $\frac{\epsilon}{d+1}$-far from being bipartite. □

Lemma 6 states that we can test the balancedness of G with parameter ϵ by testing the bipartiteness of G' with parameter $\epsilon/(d+1)$. Testing bipartiteness can be achieved by Algorithm 1, which uses random walks to find odd cycles. It was proven to be a one-sided bipartiteness tester in the bounded-degree model [21].

[3]The reduction is reminiscent of the reduction from cycle-freeness testing to bipartiteness testing in [12]. However, the reduction in [12] is randomized and between unsigned graphs, whereas our reduction is deterministic and for signed graphs.

Algorithm 1 Bipartiteness tester [21]

1: **for** $O(1/\epsilon)$ times **do**
2: Pick a node v uniformly at random.
3: Perform $\tilde{O}(\sqrt{N}/\epsilon^3)$ random walks[4] starting from v, each of length $\tilde{O}(1/\epsilon^8)$.
4: Output *reject* if a vertex u is reached by both an even-length and an odd-length path $v \rightsquigarrow u$.

It remains to prove that we can efficiently implement this tester on G'.

Lemma 7. *It is possible to implement Algorithm 1 on G' using $\mathrm{poly}(\log(N)/\epsilon)\sqrt{N}$ adjacency list queries to G.*

Proof. First note that the degree $d(u)$ of any node u can be determined using $O(\log(d(u)))$ queries using binary search. We need to be able to select a uniformly random node from G', and implement a random walk on G'. The latter is easy:

- If we are on an original node u in G' then pick a random neighbor v of u in G. If (u, v) is negative, go to v, otherwise go to the new node indexed $w^{(u,v)}$.
- If we are on a new node $w^{(u,v)}$, go to either u or v with probability $1/2$.

To select a uniformly random node from G', do the following:

(1) Pick $(u, i) \in [N] \times [d]$ uniformly at random and query for the i-th neighbor v of u in G. If u has less than i neighbors we reject.
(2) If $\sigma(u, v) = -$, with probability $1/(4d(u))$ output a random endpoint of (u, v) and terminate. Otherwise, go to next step.
(3) If $\sigma(u, v) = +$, with probability $1/4$ output $w^{(u,v)}$ and terminate. Otherwise, with probability $1/(3d(u))$ output a random endpoint of (u, v).

With probability $\geq 1/(4d)$, this scheme returns a uniformly random node from G' (and otherwise it rejects). To see this, first consider any original node u in G'. Any of its $d(u)$ incident edges is picked with an equal probability $1/(dN)$. If a negative incident edge is picked, then u is returned in step 2. with probability $1/(4d(u))$; if it is a positive incident edge then u is returned in step 3. with probability $(1 - 1/4)/(3d(u)) = 1/(4d(u))$. Hence the total probability that u is returned is

$$d(u)\frac{1}{4d(u)}\frac{1}{dN} = \frac{1}{4dN}.$$

Now consider a new node $w^{(u,v)}$ in G'. In step 1. the edge (u, v) is picked with probability $1/(dN)$, after which $w^{(u,v)}$ is returned with probability $1/4$ in step 3., yielding a total probability $1/(4dN)$. Since there are $N + m^- \geq N$ nodes in G', the total probability of returning a node is $\geq N/(4dN) = 1/(4d)$. The sampling scheme only requires a single query, and so we can sample a uniformly random node from G' using $4d \in O(1)$ queries in expectation. By Chebyshev's inequality the total number of queries will be close to its expection with high probability. □

This proves the following theorem.

[4]A single step of the random walk from a node v (with degree $d(v)$) corresponds to the following process: with probability $d(v)/d$ move to a uniformly random neighbor, and otherwise stay at v.

Theorem 8. *There exists a one-sided tester for balance in the bounded-degree model with query complexity $\tilde{O}(\sqrt{N}/\mathrm{poly}(\epsilon))$.*

Since balancedness of signed graphs generalizes bipartiteness of unsigned graphs, the $\widetilde{\Omega}(\sqrt{N})$ lower bound for testing bipartiteness in the (unsigned) bounded-degree model [22, Theorem 7.1] also applies to testing balancedness in the (signed) bounded-degree model. As a consequence, the $\sqrt{N}$-dependency of our tester is optimal.

3.2 Clusterability

In this section we prove the existence of a one-sided property tester for clusterability in the bounded-degree signed graph model. We first note that similar to the dense case we can reduce the problem to testing $O(1/\epsilon)$-clusterability. However, k-clusterability is a special case of k-colorability for unsigned graphs, and this is known not to be testable in the bounded-degree model [8], requiring $\Omega(N)$ queries. Instead, we use the forbidden subgraph characterization of clusterability by Davis [13]:

Theorem 9 ([13, Theorem 1]). *A signed graph G is clusterable if and only if G contains no cycle with exactly one negative edge.*

We call such a cycle a *bad cycle*. Algorithm 2 will try to find bad cycles by performing many random walks on the subgraph $G^+ = (V, E^+)$ induced by the positive edges E^+. Starting from a random initial node, we perform many such walks and we check for the existence of a negative edge between distinct walks. Such a negative edge will necessarily yield a bad cycle, in which case we reject the graph.

Algorithm 2 Tester for clusterability

1: **for** $O(1/\epsilon)$ times **do**
2: Pick a random node s and run `bad-cycle`(s).
3: Output *reject* if `bad-cycle`(s) = True

`bad-cycle`(s):

1: Perform $\tilde{O}(\sqrt{N}/\mathrm{poly}(\epsilon))$ random walks of length $\tilde{O}(1/\mathrm{poly}(\epsilon))$ on G^+, starting from s. Let K denote the set of all the nodes that are visited.
2: If there is a negative edge between any pair of nodes in K, then `bad-cycle`(s) ← True.

Theorem 10. *Algorithm 2 is a one-sided tester for clusterability with query complexity $\tilde{O}(\sqrt{N}/\mathrm{poly}(\epsilon))$.*

The claim about the query complexity is easy to check. The total number of random walk steps is $\tilde{O}(\sqrt{N}/\mathrm{poly}(\epsilon))$, and a single random walk step can be implemented with $\tilde{O}(1)$ queries. To check whether there exists a negative edge between any pair of nodes in K, it suffices to query the full (bounded) neighborhood of every node in K. This takes $d|K| \in \tilde{O}(\sqrt{N}/\mathrm{poly}(\epsilon))$ queries. The remainder of this section is used to prove correctness of the tester, which ultimately follows from Theorem 14.

Intuition for expanders. We first describe the intuition behind the tester. To this end, assume that there is a decomposition $V = V_1 \cup \cdots \cup V_k$ as in Figure 2 such that for each i the following holds:

(1) V_i has few positive outgoing edges:
$$|E^+(V_i, V_i^c)| \leq \frac{\epsilon}{2} d|V_i|.$$
(2) A random walk of length $\tilde{O}(1/\text{poly}(\epsilon))$ on G^+, and starting from any $s \in V_i$, ends uniformly at random inside V_i.

While such a decomposition does not generally exist, the existence of a closely related decomposition was proven in [21].

Now assume that G is ϵ-far from being clusterable. Then we claim that there must be at least $\frac{\epsilon}{2}dN$ negative edges inside the partitions $V_1, \ldots, V_k$. Indeed, if this were not the case, then we could find a valid clustering by removing these $\leq \frac{\epsilon}{2}dN$ negative edges together with the $\leq \frac{\epsilon}{2}dN$ positive edges between the partitions. This contradicts the fact that G is ϵ-far from being clusterable.

Now make the additional assumption that the number of negative edges $|E^-(V_i)|$ inside each partition V_i is $\Omega(\epsilon|V_i|)$, and consider an arbitrary node $s \in V_i$. The probability that a pair of random walks on the positive edges, starting from s, results in a bad cycle can be lower bounded by the probability that the random walk endpoints u and v form a negative edge $(u, v) \in E^-$. Since u and v are distributed uniformly, this probability is at least $|E^-(V_i)|/|V_i|^2 \in \Omega(\epsilon/N)$. Taking $\sqrt{N/\epsilon}$ independent random walks (and ignoring correlations), the total probability of finding a bad cycle then becomes $\Omega\left(\binom{\sqrt{N/\epsilon}}{2}\frac{\epsilon}{N}\right) \in \Omega(1)$.

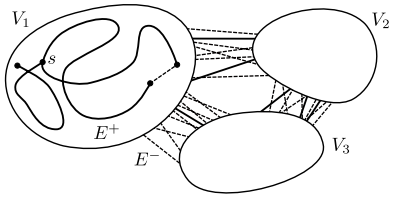


Figure 2: Signed graph decomposition. Positive edges E^+ depicted as solid lines, negative edges E^- depicted as dashed lines. G is ϵ-close to clusterable if there are $\leq \frac{\epsilon}{2}dN$ positive edges between the partitions and $\leq \frac{\epsilon}{2}dN$ negative edges inside the partitions.

General graphs. While the former section correctly captures the intuition behind the tester, the full proof of correctness is significantly more involved. We utilize various ideas from [21] regarding the graph decomposition, but also regarding bounds on the correlation between distinct random walks. The main idea of the decomposition is that for most of the vertices s in G^+, we can find a subset S so that S has few outgoing edges and a short random walk from s mixes approximately uniformly over S. We can hence set the first partition $V_1 = S$. Now we would like to repeat the argument for the remaining graph $H^+ = G^+[V - V_1]$, induced on the node subset $V - V_1$. The problem is that the next subset $S' \subseteq V - V_1$ will be "good" for random walks in H^+, but these can behave very differently from the original random walks in G^+. This problem is dealt with by defining a Markov chain $M(H^+)$ on the unpartitioned subgraph H^+ such that (i) we can use $M(H^+)$ to cut off a new partition S' from H^+, but also (ii) that the behavior of walks according to $M(H^+)$ is related to the behavior of the original random walks in G^+. Details of the Markov chain $M(H^+)$ are given in Appendix B.1. Lemma 11 is the key lemma in the graph decomposition of [21]. Here $q_{s,v}(t)$ denotes the probability that after t steps of the Markov chain $M(H^+)$ starting from s we end in v.

Lemma 11 ([21, Corollary 3 and Lemma 4.3]). *Let H^+ be a subgraph of G^+ with at least $\epsilon N/4$ vertices. Then for at least half of the vertices s in H^+ there exists a subset of vertices S in H^+, a value $\beta \in \widetilde{\Omega}(\epsilon^2)$ and an integer $t \in \tilde{O}(1/\epsilon^3)$ such that*

(1) *The number of edges between S and the rest of H^+ is at most $\epsilon d|S|/2$.*
(2) *For every $v \in S$ it holds that $\sqrt{\frac{\beta}{|S||H^+|}} \leq q_{s,v}(t) \leq \frac{1}{\epsilon}\sqrt{\frac{\beta}{|S||H^+|}}$.*

Under these conditions, we can prove that if the original signed graph G has many negative edges inside the subset S then the modified Markov chain $M(H^+)$ will find a bad cycle. Lemma 12 is new, and we prove it in Appendix C.2.

Lemma 12. *Let H^+ be a subgraph of G^+, s a vertex in H^+ and S a subset of vertices in H^+. Assume that there exist $\alpha > 0$, $F \geq 1$, t such that $\alpha \leq q_{s,v}(t) \leq F\alpha$ for every $v \in S$. If the original graph G has at least $\epsilon d|S|/2$ negative edges inside S then $m \in \Omega\left(\frac{F}{\epsilon\alpha\sqrt{|S|}}\right)$ runs of $M(H^+)$ over t steps and starting from s will return a bad cycle with probability at least 0.99.*

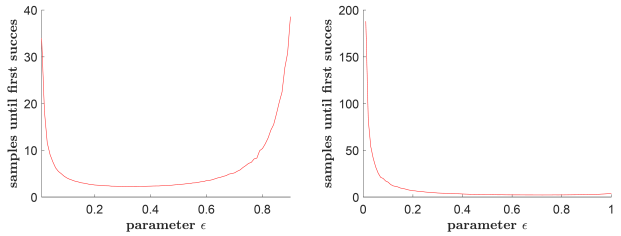
Ultimately we are of course interested in the behavior of random walks in G^+, rather than that of $M(H^+)$. The following statement (Appendix C.1) shows that both are closely related.

Claim 13. *Assume that there exists s and m such that m walks of $M(H^+)$ of length $\tilde{O}(1/\epsilon^3)$ and starting from s result in a bad cycle w.p. $\geq$ 0.99. Then m random walks in G^+ of length $\tilde{O}(1/\epsilon^8)$ and starting from s will also result in a bad cycle w.p. $\geq$ 0.99.*

Algorithm correctness. Instead of proving that if G is ϵ-far from clusterable we reject, we prove the contrapositive: if G is accepted with large probability, then G must be close to being clusterable.

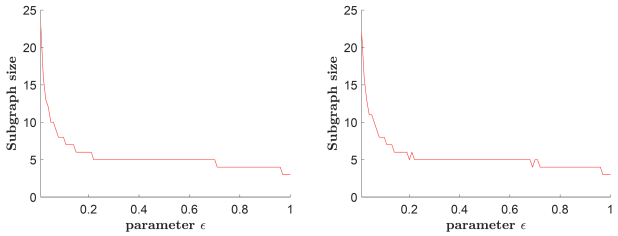
Theorem 14. *If Algorithm 2 accepts a graph G with probability greater than 1/3, then G must be 2ϵ-close to being clusterable.*

Proof. To prove this, let G be a graph that is accepted with probability greater than 1/3. We say that a vertex s is *good* if `bad-cycle`(s) in Algorithm 2 returns a bad cycle with probability at most 0.1. Otherwise it is *bad*. Since we reject with probability less than 2/3, and we consider $\Omega(1/\epsilon)$ starting vertices, there can be only $\epsilon N/16$ bad vertices (for the appropriate constant in the $\Omega(\cdot)$ notation). We show that under these circumstances we can find a valid clustering by removing less than $2\epsilon dN$ edges. To this end, we iteratively separate a subset S that has at most $\epsilon d|S|/2$ positive outgoing edges and at most $\epsilon d|S|/2$ negative internal edges. We call such a subset an *ϵ-good cluster*. At a given step, let H^+ denote the unpartitioned graph. We wish to invoke Lemma 11. Call a vertex s for which the lemma holds a "useful" vertex with respect to H^+. While $|H^+| \geq \epsilon N/4$, the lemma ensures that there are $\geq \epsilon N/8$ useful vertices. Since there are at most $\epsilon N/16$ bad vertices, this implies that there exists a vertex s that is both good and useful. By Lemma 11 there exists a subset S in H^+ that has at most $\frac{\epsilon}{2}d|S|$ (positive) edges to the rest of H^+. Moreover, by Lemma 12 and Claim 13, the set S is such that if the original signed graph G has at least $\epsilon d|S|$



(a) All positive initial graph (b) $(N/10)$–clustered initial graph

Figure 3: Empirical query complexity for testing signed triangle freeness in the dense model on two types of synthetically constructed graphs.



(a) All positive initial graph (b) 2–clustered initial graph

Figure 4: Empirical sampling complexity for testing balance in the dense model on two types of synthetically constructed graphs.

negative edges inside S then `bad-cycle`(s) will return a bad cycle with probability at least 0.99. However, we assumed that s is a good vertex and so the latter probability can be at most 0.1. This implies that G must have less than $\epsilon d|S|/2$ negative edges inside S, and hence S is an ϵ-good cluster.

We repeat this process until $|H^+| < \epsilon N/4$. If $V_1, \ldots, V_k$ denote the ϵ-good clusters that we have cut off, then we have a partition $V = V_1 \cup \cdots \cup V_k \cup H$ such that the number of positive edges between the partitions is at most

$$d|H| + \sum_i |E^+(V_i, V_i^c)| \leq \frac{\epsilon}{4} dN + \sum_i \frac{\epsilon}{2} d|V_i| < \epsilon dN,$$

and the number of negative edges inside the partitions is at most

$$d|H| + \sum_i |E^-(V_i)| \leq \frac{\epsilon}{4} dN + \sum_i \frac{\epsilon}{2} d|V_i| < \epsilon dN.$$

Removing these less than $2\epsilon dN$ edges yields a valid clustering, so that G must be 2ϵ-close to clusterable. This proves Theorem 14. □

4 EXPERIMENTS

All experiments are performed on an Intel Core i5 machine at 1.8 GHz with 16 GB RAM and the methods are publicly available.[5]

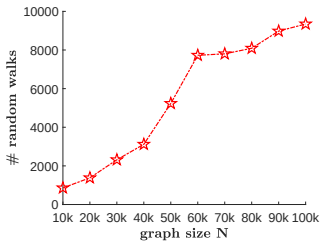
4.1 Query complexity

We experimentally verify the query complexity of three of our testers in a more practical scenario. For example, the theoretical query complexity of testing signed triangle freeness in the dense graph model is a tower function of height $\log(1/\epsilon)$. We experimentally verify if such a query complexity is actually necessary in realistic networks, or if a smaller number of queries is sufficient.

Signed triangle freeness. We start by testing the practical dependency on the parameter ϵ of our signed triangle freeness tester in the dense model. The triangle sign configuration of interest will be $\{+,+,-\}$. First, we aim to generate synthetic complete graphs that are within a controllable distance to a graph that does not have any such triangles. In order to achieve this, we start from an initial complete graph that has no such triangles. Then we randomly switch the sign of an ϵ-fraction of the edges, in the hope that the distance of the resulting graph to being signed triangle free is roughly equal to the number of signs that we have switched. Of course, this will not always be true, since the effect of multiple sign switches can nullify each other or if the switch induces no new triangles. However, for small ϵ it is a reasonably good approximation for certain types of graphs. This is more likely to be true if the graph has a good amount of positive edges, which has been observed in many real-life networks, where often more than 80% of the interactions are positive [15, 30]. In such a case, a random flip will most likely turn a positive edge into a negative one, and induce several $\{+,+,-\}$ triangles. After switching, we empirically verify how many vertex triples we need to sample until we detect a triangle. The smaller ϵ will be, the less amount of such triangles that will be present in the graph, and the detection should be harder. Estimating the expected number of required samples until triangle detection is done by noticing that this is a geometric distribution with expectation $1/p_t$, where p_t is the probability that a randomly chosen triplet t is indeed a $\{+,+,-\}$ triangle. The quantity p_t is estimated by sampling a pool of $25k$ random triplets, and returning the relative fraction in the pool that induce a triangle.

We consider two types of initial graphs that have no $\{+,+,-\}$ triangles; (a) a complete graph with only positive edges, and (b) a complete signed graph that is clusterable into a constant number of clusters, all with equal cluster size $N/10$. The latter models networks with cluster structure. For both these types of graphs, randomly switching an ϵ-fraction of the signs should induce a graph that is $O(\epsilon)$-far from being signed triangle free, provided that ϵ is not too large. We set $N = 10k$ and $\epsilon \in \{0.01, 0.02, \ldots, 1\}$. Figure 3 shows the results. In Figure 3a the initial graph is a complete signed graph with all positive edges. The increase of the required number of samples to detect a triangle for large ϵ is an artefact of our experimental construct. Indeed, as discussed before, sign switches might nullify each other. E.g., for $\epsilon = 1$ the resulting graph is switched to the all-negative graph and thus contains no $\{+,+,-\}$ triangles. This blow-up is observed in the tail of the plot. Figure 3b shows the result for an initial graph with cluster structure and a fixed number of clusters. In this case switching the sign of all the edges ($\epsilon = 1$) does not give a graph without triangles. For ϵ approaching one the number of required samples does increase again slightly, but does not blow up. For small ϵ we observe that we need to sample more triplets to encounter a triangle as compared to Figure 3a. In both cases, we also remark that the number of vertices N has

[5] https://tinyurl.com/vsed62kz



(a) All positive initial d-regular graph, switching ϵdN random edges to negative with $\epsilon = 0.15$ fixed.

Figure 5: Empirical query complexity for testing clusterability in the bounded degree model.

negligible influence on the results. Both settings also indicate a dependency behaving more similar to $O(1/\text{poly}(\epsilon))$, as opposed to the theoretical $O(\text{tower}(\log(1/\epsilon)))$.

Balance. Secondly, we test the effectiveness of testing balance in the dense model. The tester samples $\tilde{O}(1/\epsilon)$ vertices, queries the whole induced subgraph (leading to a $\tilde{O}(1/\epsilon^2)$ query complexity), and accepts if and only if the induced subgraph is balanced. We generate graphs similar as in testing signed triangle freeness, but now the second type of graphs is clustered into exactly two clusters. Edges inside clusters are positive and across they are negative. Set $N = 10k$ and we let $\epsilon \in \{0.01, 0.02, \ldots, 1\}$. We randomly switch an ϵ-fraction of the edges. For each value of ϵ, we then incrementally try to find the size of the smallest induced subgraph that correctly rejects in 2/3 of the cases over a sample size of 5k subgraphs. Figure 4a shows the results when the initial graph is a complete graph with all positive edges. Figure 4b shows the results when the initial graph is a balanced graph with two clusters of equal size. Interestingly, the results are nearly identical.

Clusterability. Finally, we test Algorithm 2 for testing clusterability in the bounded-degree model. We check the query complexity's dependency on N, for fixed $\epsilon = 0.15$. We generate asymptotically uniformly random sparse d-regular graphs of varying sizes $N \in \{10k, 20k, \ldots, 100k\}$, with fixed $d = 4$, using the method from [24]. The initial graphs all have positive edges. For each randomly generated graph, we switch the signs of ϵdN randomly chosen edges from positive to negative. Again the intuition is that for this relatively small amount of switches, the resulting graph should be ϵ-far from being clusterable. The question is if the detection of a bad cycle gets more difficult as N increases. We implement the `bad-cycle` routine from Algorithm 2, and we check how many fixed length random walks (we set the length to 10) we need to start from a randomly chosen node such that there is a negative edge amongst the encountered nodes in the walks. We take the average over 10 repeats. Figure 5 shows that the results are in accordance with the theoretical $\sqrt{N}$-dependency.

4.2 Insights on real datasets

In a final experiment, we run the balance tester in the dense model on five real-life networks in Table 2. For each dataset, we list the spectral signed frustration index ϕ, which is an approximation of the signed frustration index based on the smallest eigenvalue of the signed Laplacian matrix [29]. We select three low/medium/high values of $\epsilon \in \{\phi/10, \phi, 10\phi\}$ and interpret the results. We sample $2/\epsilon$ vertices,[6] query the induced subgraph and output accept/reject if and only if this subgraph it is balanced. Table 3 shows the majority observation over 10 repetitions. The number between brackets is the amount of times the winning outcome was observed.

Table 2: Overview of the real-life datasets. The statistic ϕ denotes the spectral signed frustration index.

Data	$\lvert V\rvert$	$\lvert E^+\rvert$	$\lvert E^-\rvert$	ϕ
Cloister [29]	18	97	87	0.1533
Tribes [29]	16	29	29	0.0358
Bitcoin OTC [28, 31]	5,881	32,029	3,563	0.0048
Bitcoin Alpha [28, 29]	3,783	22,651	1,536	0.0047
Wiki Elections [31]	7,118	81,318	22,357	0.0014

Table 3: Output of the balance testing algorithm. A checkmark (✓) denotes an accept, a crossmark (×) denotes a reject.

Balance test results	$\epsilon = \phi/10$	$\epsilon = \phi$	$\epsilon = 10\phi$
Cloister [29]	× (10)	× (10)	✓ (9)
Tribes [29]	× (10)	× (10)	✓ (8)
Bitcoin OTC [28, 31]	× (8)	✓ (10)	✓ (10)
Bitcoin Alpha [28, 29]	× (8)	✓ (9)	✓ (10)
Wiki Elections [31]	× (8)	✓ (6)	✓ (10)

What do we learn from this? Recall the definition of a property tester in Section 1.1. The contrapositive of a one-sided tester implies two things; (a) if we reject, then we learn that the graph does not have the property and (b) if we accept with probability greater than 1/3, then we learn that the graph is ϵ-close to having the property. So the question is whether we learned false conclusions when running this experiment (e.g., due to a bad choice of constants in the sampling, or due to the density of the graphs). Outputting reject is never a wrong answer, since all networks are not balanced. More importantly, the algorithm correctly rejected in most trials whenever $\epsilon = \phi/10$, which is clearly the desired outcome. Moreover, the algorithm accepted most graphs whenever $\epsilon = 10\phi$, which does not contradict the interpretation of ϕ. Naturally, the denser a graph the more accurately the notion of distance in the dense model corresponds to the actual spectral signed frustration index ϕ.

ACKNOWLEDGMENTS

This work has benefited from discussions with Jop Briët, Aristides Gionis, Oded Goldreich and Christian Sohler. Florian Adriaens is currently supported by Helsinki Institute for Information Technology HIIT, and while at KTH supported by the ERC Advanced Grant REBOUND (834862), the EC H2020 RIA project SoBigData (871042), and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

[6]The factor two is arbitrary and can be increased to improve the reliability of the results.

REFERENCES

[1] Nir Ailon, Moses Charikar, and Alantha Newman. 2008. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)* 55, 5 (2008), 1–27.

[2] Noga Alon, W.Fernandez de la Vega, Ravi Kannan, and Marek Karpinski. 2003. Random sampling and approximation of MAX-CSPs. *J. Comput. System Sci.* 67, 2 (2003), 212 – 243. STOC 2002.

[3] Noga Alon and Michael Krivelevich. 2002. Testing k-Colorability. *SIAM J. Discret. Math.* 15, 2 (Feb. 2002), 211–227. https://doi.org/10.1137/S0895480199358655

[4] Noga Alon and Asaf Shapira. 2008. Every monotone graph property is testable. *SIAM J. Comput.* 38, 2 (2008), 505–522.

[5] N. Alon and J. H. Spencer. 2004. *The probabilistic method.* John Wiley & Sons.

[6] G. Andersson and L. Engebretsen. 2002. Property testers for dense constraint satisfaction programs on finite domains. *Random Structures & Algorithms* (2002).

[7] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. 2004. Correlation clustering. *Machine learning* 56, 1 (2004), 89–113.

[8] Andrej Bogdanov, Kenji Obata, and Luca Trevisan. 2002. A lower bound for testing 3-colorability in bounded-degree graphs. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 93–102.

[9] Francesco Bonchi, Edoardo Galimberti, Aristides Gionis, Bruno Ordozgoiti, and Giancarlo Ruffo. 2019. Discovering Polarized Communities in Signed Networks. In *28th ACM Conference on Information and Knowledge Management (CIKM '19)*. Association for Computing Machinery, New York, NY, USA.

[10] D. Cartwright and F. Harary. 1956. Structural balance: a generalization of Heider's theory. *Psychological review* 63 5 (1956), 277–293.

[11] Kai-Yang Chiang, Nagarajan Natarajan, Ambuj Tewari, and Inderjit Dhillon. 2011. Exploiting Longer Cycles for Link Prediction in Signed Networks. *International Conference on Information and Knowledge Management, Proceedings*, 1157–1162.

[12] Artur Czumaj, Oded Goldreich, Dana Ron, C Seshadhri, Asaf Shapira, and Christian Sohler. 2014. Finding cycles and trees in sublinear time. *Random Structures & Algorithms* 45, 2 (2014), 139–184.

[13] James A Davis. 1967. Clustering and structural balance in graphs. *Human relations* 20, 2 (1967), 181–187.

[14] Erik D Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. 2006. Correlation clustering in general weighted graphs. *Theoretical Computer Science* 361, 2-3 (2006), 172–187.

[15] Tyler Derr, Charu Aggarwal, and Jiliang Tang. 2018. Signed Network Modeling Based on Structural Balance Theory. In *27th ACM International Conference on Information and Knowledge Management* (Torino, Italy) *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 557–566.

[16] Eldar Fischer and Ilan Newman. 2007. Testing versus estimation of graph properties. *SIAM J. Comput.* 37 (2007).

[17] Jacob Fox. 2011. A new proof of the graph removal lemma. *Annals of Mathematics* (2011), 561–579.

[18] Oded Goldreich. 2010. Introduction to testing graph properties. In *Property testing*. Springer, 105–141.

[19] Oded Goldreich. 2017. *Testing Graph Properties in the Dense Graph Model.* Cambridge University Press, 162–212. https://doi.org/10.1017/9781108135252.010

[20] Oded Goldreich, Shafi Goldwasser, and Dana Ron. 1998. Property Testing and Its Connection to Learning and Approximation. *J. ACM* 45, 4 (July 1998), 653–750. https://doi.org/10.1145/285055.285060

[21] Oded Goldreich and Dana Ron. 1999. A sublinear bipartiteness tester for bounded degree graphs. *Combinatorica* 19, 3 (1999), 335–373.

[22] Oded Goldreich and Dana Ron. 2004. Property Testing in Bounded Degree Graphs. *Algorithmica* 32 (01 2004). https://doi.org/10.1145/258533.258627

[23] Frank Harary. 1953. On the notion of balance of a signed graph. *Michigan Math. J.* 2, 2 (1953), 143–146. https://doi.org/10.1307/mmj/1028989917

[24] Jeong Han Kim and Van H Vu. 2003. Generating random regular graphs. In *Thirty-fifth annual ACM symposium on Theory of computing*.

[25] D. König. 1936. *Theorie der endlichen und unendlichen Graphen: Kombinatorische Topologie der Streckenkomplexe.* Vol. 16. Akademische Verlagsgesellschaft.

[26] Akash Kumar, C Seshadhri, and Andrew Stolman. 2019. Random walks and forbidden minors II: a $\mathrm{poly}(d\varepsilon^{-1})$-query tester for minor-closed properties of bounded degree graphs. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. 559–567.

[27] Akash Kumar, C Seshadhri, and Andrew Stolman. 2020. Random Walks and Forbidden Minors I: An $n^{1/2+o(1)}$-Query One-Sided Tester for Minor Closed Properties on Bounded Degree Graphs. *SIAM J. Comput.* (2020).

[28] Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. 2016. Edge weight prediction in weighted signed networks. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 221–230.

[29] Jerome Kunegis. 2013. KONECT: The Koblenz Network Collection. (2013).

[30] J. Leskovec, D. Huttenlocher, and J. Kleinberg. 2010. Predicting Positive and Negative Links in Online Social Networks. In *19th International Conference on World Wide Web*. Association for Computing Machinery, NY, USA.

[31] J. Leskovec and R. Sosič. 2016. SNAP: A General-Purpose Network Analysis and Graph-Mining Library. *ACM (TIST)* (2016).

[32] A. Mara, Y. Mashayekhi, J. Lijffijt, and T. de Bie. 2020. CSNE: Conditional Signed Network Embedding. In *29th ACM International Conference on Information and Knowledge Management (CIKM '20)*. ACM.

[33] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. 2006. Tolerant property testing and distance approximation. *J. Comput. System Sci.* (2006).

[34] I. Ruzsa and E. Szemerédi. 1978. Triple systems with no six points carrying three triangles. *Combinatorics, Coll. Math. Soc. J. Bolyai* (1978).

[35] C. Sohler. 2012. Almost optimal canonical property testers for satisfiability. In *IEEE 53rd Annual Symposium on Foundations of Computer Science*. IEEE.

[36] J. Tang, Yi Chang, C. Aggarwal, and H. Liu. 2016. A survey of signed network mining in social media. *ACM Computing Surveys (CSUR)* 49, 3 (2016).

[37] R.C. Tzeng, B. Ordozgoiti, and A. Gionis. 2020. Discovering conflicting groups in signed networks. *Advances in Neural Information Processing Systems* (2020).

[38] Nate Veldt. 2022. Correlation Clustering via Strong Triadic Closure Labeling: Fast Approximation Algorithms and Practical Lower Bounds. In *International Conference on Machine Learning*. PMLR, 22060–22083.

[39] T. Zaslavsky. 2018. Negative (and positive) circles in signed graphs: A problem collection. *AKCE Journal of Graphs and Combinatorics* 15, 1 (2018).

A SIGNED TRIANGLE FREENESS

A.1 Dense model

Testing triangle freeness for unsigned graphs has been well studied. It is a direct application of the triangle removal lemma [17, 34]. For some function f, the canonical one-sided tester simply picks $f(\epsilon)$ random triples in $[N]$ and rejects the graph if any of the triples induces a triangle. If the graph is triangle free, then we always accept the graph, so that the tester is indeed one-sided. If the graph is ϵ-far from being triangle free, than the triangle removal lemma of Fox [17] proves that the graph contains at least $\delta(\epsilon)\binom{n}{3}$ triangles. Here $\delta(\epsilon)$ is a function bounded by the inverse of the towering function $\text{tower}(\log(1/\epsilon))$, which corresponds to a tower of 2's of height $O(\log(1/\epsilon))$ (i.e., 2-to-the-2-to-the-...-to-the-2, $O(\log(1/\epsilon)$ times). Hence if we sample $f(\epsilon) \in \Theta(1/\delta(\epsilon))$ triples, then with high probability one of them will induce a triangle and we will correctly reject the graph. This yields a total query complexity of $O(1/\delta(\epsilon))$.

Testing signed triangle freeness in signed graphs can be analyzed in a very similar way. By interpreting the edge signs of a graph as an edge-coloring, we can use Fox's *colored* triangle removal lemma [17]. This lemma states that if an edge-colored graph is ϵ-far from being free of a certain colored triangle, then the graph contains at least $\delta'(\epsilon)\binom{n}{3}$ such induced triangles, where $\delta'(\epsilon)$ is again bounded by the inverse of $\text{tower}(\log(1/\epsilon))$. By the same argument as in the unsigned case, this implies the existence of a one-sided tester for colored (or signed) triangle freeness in the dense graph model with query complexity $O(\text{tower}(\log(1/\epsilon)))$. This proves Theorem 1.

A.2 Bounded-degree model

Testing triangle freeness in the bounded-degree (unsigned) graph model is significantly easier to analyze than doing so for the dense model [18]. In fact, the exact same argument applies to signed graphs and we describe it here for completeness. Given query access to a signed graph, consider the following testing algorithm: pick $\tilde{O}(1/\epsilon)$ nodes and reject if any of them is part of a signed triangle. We can check whether a node is part of a signed triangle simply by querying for its neighbors, and the neighbors of its neighbors. In the bounded-degree model this takes only $\tilde{O}(1)$ queries. If the graph is signed triangle free then we will never reject. On the other hand, if at least ϵdN edges have to removed in order to make the graph signed triangle free, then at least ϵN nodes must be part of a signed triangle. With large probability the algorithm will sample such a node and consequently reject the graph. This proves Theorem 5.

B TECHNICAL DETAILS FOR TESTING CLUSTERABILITY IN THE DENSE MODEL

B.1 Clusterability

The following theorem is proven by Andersson and Engebretsen [6].

Theorem 15 ([6, Theorem 2]). *Consider a constraint family $\mathcal{F} = \{f : D^\ell \to \{0,1\}\}$ over ℓ variables in domain D, and let Σ denote the maximum number of constraints that can be simultaneously satisfied. An ℓ-CSP-D instance $\mathcal{I}$ over n variables with constraint family $\mathcal{F}$ is described by a collection of constraints $\{(f, x_{i_1}, \dots, x_{i_\ell})\}$ with $f \in \mathcal{F}$ and $i_1, \dots, i_\ell \in [n]$. It is possible to approximate the maximum number of satisfiable constraints $\max(\mathcal{I})$ up to error ϵn^k with probability at least $1-\delta$ using*

$$\tilde{O}\left(\frac{|\mathcal{F}|\Sigma^7\ell^2}{\epsilon^7}\right)$$

queries[7] and time $\exp\left(\tilde{O}\left(\frac{\Sigma^3\ell}{\epsilon^3}\right)\right)$.

The problem of k-clusterability is a special instance of this problem. Define the family $\mathcal{F} = \{f^+, f^- : [k]^2 \to \{0,1\}\}$ by $f^+(x,y) = 1$ if $x = y$ and 0 otherwise, and $f^-(x,y) = 0$ if $x = y$ and 1 otherwise. Now given a signed graph G, we can define a collection $\mathcal{I}_G$ of constraints by adding constraint (f^+, x, y) if (x, y) is a positive edge, and (f^-, x, y) if (x, y) is a negative edge. We can query $\mathcal{I}_G$ using a single query to the adjacency matrix of G. Moreover, the k-frustration index of G is given by

$$|E(G)| - \max(\mathcal{I}_G),$$

with $|E(G)|$ the number of edges in G. Using that $\Sigma = 1$, $|\mathcal{F}| = 2$ and $\ell = 2$ for the family $\mathcal{F}$, it follows from Theorem 15 that we can find an ϵN^2 approximation of $\max(\mathcal{I}_G)$ using $\tilde{O}(1/\epsilon^7)$ queries and time $\exp(\tilde{O}(1/\epsilon^3))$. In addition we can easily approximate $|E(G)|$ to additive error ϵN^2 by randomly sampling entries of the adjacency matrix of G. Combining these gives an approximation algorithm for the k-frustration index with additive error ϵN^2, and hence a tolerant tester for k-clusterability.

C TECHNICAL DETAILS FOR TESTING CLUSTERABILITY IN THE BOUNDED-DEGREE MODEL

C.1 Modified Markov chain

In this section we describe the technical details on the modified Markov chain applied to our setting. This Markov chain was initially proposed by in [21] in the context of unsigned graphs. Let $G = (V, E, \sigma)$ be a signed graph and let $G^+ = (V, E^+)$ be the subgraph induced on the positive edge set. Let H^+ be a subgraph of G^+, and let ℓ_1, ℓ_2 be integers. The boundary $B(H^+)$ of H^+ consists of those vertices in H^+ that have an edge in G^+ that leaves H^+. Let $\hat{H}^+$ be the graph obtained by appending to every boundary node $v \in B(H^+)$ an *auxiliary path* of length ℓ_1 with node set $a_{v,1}, \dots, a_{v,\ell_1}$.

We define a surjective mapping ϕ from random walks W in G^+ of length $L = \ell_1\ell_2$ to walks $\phi(W)$ in $\hat{H}^+$ of length ℓ_1. If $W = v_0, \dots, v_L$, let $i_0, \dots, i_k$ be the timesteps for which $v_{i_j} \in H^+$. The mapping is defined essentially by contracting all length-$(< \ell_2)$ walks outside of H^+, and routing any length-$(\geq \ell_2)$ walk outside of H^+ onto an auxiliary path. More precisely:

- *Contract:* If W does not perform ℓ_2 or more consecutive steps outside of H^+ before it made ℓ_1 steps (in total) in H^+, then
$$\phi(W) = v_{i_0}, \dots, v_{i_{\ell_1}}.$$
- *Contract and route:* In the other case, let i_r be the first index that precedes a walk of $\geq \ell_2$ consecutive steps outside of H^+. Then
$$\phi(W) = v_{i_0}, \dots, v_{i_r}, a_{v_{i_r},1}, \dots, a_{v_{i_r},\ell_1 - i_r}.$$

[7] A query to the instance $\mathcal{I}$ takes the form $Q = (f, x_{i_1}, \dots, x_{i_\ell})$ and returns 1 if $Q \in \mathcal{I}$ and 0 otherwise.

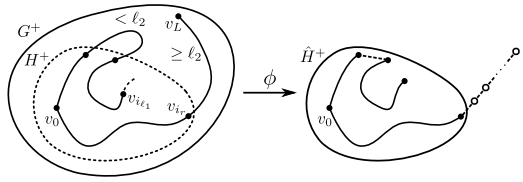


Figure 6: Illustration of mapping ϕ from walks of length $L = \ell_1\ell_2$ on G^+ to walks of length ℓ_1 on $\hat{H}^+$.

The distribution $\Pr_G(W)$ over length-L walks W in G^+ induces a distribution $\Pr_M(U)$ over length-ℓ_1 walks U in $\hat{H}^+$ by setting

$$\Pr_M(U) = \sum_{W:\phi(W)=U} \Pr_{G^+}(W).$$

We now define a Markov chain $M(H^+)$ on $\hat{H}^+$ such that length-ℓ_1 walks U of $M(H^+)$ have the same distribution $\Pr_M(U)$. In the definition of $M(H^+)$ we use the quantity $p^H_{v,u}(t)$ for $v, u \in H^+$, which denotes the probability that a random walk from v will take $t-1$ steps outside of H^+ and end in u at the t-th step. The Markov chain $M(H^+)$ is defined as follows:

- For every $v, u \in H^+$: $q_{v,u} = \sum_{t=1}^{\ell_2-1} p^H_{v,u}(t)$.
- For every $v \in B(H^+)$:
  - $q_{v,a_{v,1}} = \sum_{u\in H}\sum_{t\geq\ell_1} p^H_{v,u}(t)$,
  - for every ℓ, $1 \leq \ell < \ell_1$, $q_{a_{v,\ell},a_{v,\ell+1}} = 1$,
  - for every $u \in H$, $q_{a_{v,\ell_1},u} = q^{-1}_{v,a_{v,1}} \sum_{t\geq\ell_2} p^H_{v,u}(t)$.

Claim 13 states that if we find a bad cycle with the modified Markov chain, then we will also find a bad cycle using the original random walk. We say that a set of walks *results in a bad cycle* if the original graph G has a negative edge between two distinct vertices of the walks.

Proof of Claim 13. Let U_1 and U_2 denote length-t walks of $M(H^+)$ that result in a bad cycle. If W_1 and W_2 are length-L walks on G^+ with $\phi(W_1) = U_1$ and $\phi(W_2) = U_2$, then W_1 and W_2 will also result in a bad cycle. Now let $\mathbb{1}_{\text{bad}}(X_1,\dots,X_m)$ denote the indicator of whether the walks $X_1,\dots,X_m$ (in G^+ or $M(H^+)$) form a bad cycle. By our former remark we know that $\mathbb{1}_{\text{bad}}(W_1,\dots,W_m) \geq \mathbb{1}_{\text{bad}}(\phi(W_1),\dots,\phi(W_m))$. For notational convenience, define $Z(W) = \Pr_{G^+}(W_1)\dots\Pr_{G^+}(W_m)$. We can lower bound the probability that m walks in G^+ form a bad cycle:

$$\begin{aligned}
&\sum_{W_1,\dots,W_m} \Pr_{G^+}(W_1)\dots\Pr_{G^+}(W_m)\mathbb{1}_{\text{bad}}(W_1,\dots,W_m)\\
&= \sum_{U_1,\dots,U_m}\left(\sum_{W_1:\phi(W_1)=U_1}\cdots\sum_{W_m:\phi(W_m)=U_m} Z(W)\cdot\mathbb{1}_{\text{bad}}(W_1,\dots,W_m)\right)\\
&\geq \sum_{U_1,\dots,U_m}\left(\sum_{W_1:\phi(W_1)=U_1}\cdots\sum_{W_m:\phi(W_m)=U_m} Z(W)\cdot\mathbb{1}_{\text{bad}}(U_1,\dots,U_m)\right)\\
&= \sum_{U_1,\dots,U_m} \Pr_M(U_1)\dots\Pr_M(U_m)\mathbb{1}_{\text{bad}}(U_1,\dots,U_m)\\
&\geq 0.99,
\end{aligned}$$

proving the claim. □

C.2 Sufficient condition for bad cycle

Proof of Lemma 12. For $1 \leq i, j \leq m$, let $\eta_{i,j}$ be the random variable so that $\eta_{i,j} = 1$ if the i-th and j-th walk form a bad cycle and otherwise $\eta_{i,j} = 0$. We now bound the probability that we do not find a bad cycle, which is $\Pr(\sum_{i<j}\eta_{i,j} = 0)$. To this end we use the bound

$$\Pr\left[\sum_{i<j}\eta_{i,j} = 0\right] \leq \frac{\operatorname{Var}\left[\sum_{i<j}\eta_{i,j}\right]}{\mathbb{E}[\sum_{i<j}\eta_{i,j}]^2} \tag{1}$$

which can be derived from Chebyshev's inequality [5, Theorem 4.3.1]. To bound $\operatorname{Var}\left[\sum_{i<j}\eta_{i,j}\right]$, we define $\bar{\eta}_{i,j} = \eta_{i,j} - \mathbb{E}[\eta_{i,j}]$. By [21, Equation (20)] we can then rewrite

$$\operatorname{Var}\left[\sum_{i<j}\eta_{i,j}\right] = \mathbb{E}\left[\left(\sum_{i<j}\bar{\eta}_{i,j}\right)^2\right] = \binom{m}{2}\mathbb{E}\left[\bar{\eta}^2_{1,2}\right] + 4\binom{m}{3}\mathbb{E}\left[\bar{\eta}_{1,2}\bar{\eta}_{2,3}\right].$$

The first term is bounded by $\mathbb{E}\left[\bar{\eta}^2_{1,2}\right] \leq \mathbb{E}\left[\eta^2_{1,2}\right] = \mathbb{E}\left[\eta_{1,2}\right]$. Now we bound the second term, where we let v_i denote the endpoint of the i-th walk:

$$\begin{aligned}
\mathbb{E}\left[\bar{\eta}_{1,2}\bar{\eta}_{2,3}\right] &\leq \mathbb{E}\left[\eta_{1,2}\eta_{2,3}\right] = \Pr\left[\eta_{1,2} = 1 \text{ and } \eta_{2,3} = 1\right]\\
&= \Pr\left[\eta_{1,2} = 1 \text{ and } (v_2, v_3) \in E^-\right]\\
&= \sum_u \Pr\left[\eta_{1,2} = 1 \text{ and } v_2 = u \text{ and } (u, v_3) \in E^-\right]\\
&= \sum_u \Pr\left[(u, v_3) \in E^-\right]\cdot\Pr\left[\eta_{1,2} = 1 \text{ and } v_2 = u\right]\\
&= \sum_u\left(\sum_{w:(u,w)\in E^-}\Pr\left[v_3 = w\right]\right)\cdot\Pr\left[\eta_{1,2} = 1 \text{ and } v_2 = u\right]\\
&\leq \left(d\max_w \Pr[v_3 = w]\right)\sum_u \Pr\left[\eta_{1,2} = 1 \text{ and } v_2 = u\right]\\
&= \left(d\max_w \Pr[v_3 = w]\right)\mathbb{E}[\eta_{1,2}]\\
&\leq dF\alpha\,\mathbb{E}[\eta_{1,2}].
\end{aligned}$$

If we denote $\gamma = \mathbb{E}[\eta_{1,2}]$ then we get the bound $\operatorname{Var}\left[\sum_{i<j}\eta_{i,j}\right] \in O(m^2\gamma + m^3dF\alpha\gamma)$. Combined with (1) this gives the bound

$$\Pr\left[\sum_{i<j}\eta_{i,j} = 0\right] \in O\left(\frac{m^2\gamma + m^3dF\alpha\gamma}{m^4\gamma^2}\right).$$

If we let E^-_S denote the set of negative edges with both endpoints inside S then we can lower bound

$$\gamma = \sum_{(u,v)\in E^-_S} q_{s,u}(t)q_{s,v}(t) \geq \alpha^2|E^-_S|.$$

Combined with the fact that $m \in \Omega\left(\frac{F}{\epsilon\alpha\sqrt{|S|}}\right)$ this gives the bound $\Pr\left[\sum_{i<j}\eta_{i,j} = 0\right] \in O(\epsilon/(dF^2) + 1/\sqrt{|S|})$, which is ≤ 0.01 for the right choice of constants. □