Applied Psychological Measurement
2021, Vol. 45(4) 253–267
© The Author(s) 2021
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/01466216211013902
journals.sagepub.com/home/apm
SAGE

Graph Theory Approach to Detect Examinees Involved in Test Collusion

Dmitry I. Belov[1], and James A. Wollack[2]

Abstract

Test collusion (TC) is sharing of test materials or answers to test questions before or during the test (important special case of TC is item preknowledge). Because of potentially large advantages for examinees involved, TC poses a serious threat to the validity of score interpretations. The proposed approach applies graph theory methodology to response similarity analyses for identifying groups of examinees involved in TC without using any knowledge about parts of test that were affected by TC. The approach supports different response similarity indices (specific to a particular type of TC) and different types of groups (connected components, cliques, or near-cliques). A comparison with an up-to-date method using real and simulated data is presented. Possible extensions and practical recommendations are given.

Keywords

test security, test collusion, item preknowledge, graph theory, maximum clique problem, hypothesis testing, Markov chain Monte Carlo

Introduction

Test collusion (TC) is sharing of test materials or answers to test questions before or during the test. There are many potential sources of shared information, including teachers, test preparation entities, the internet, or even examinees collaborating during the exam. Because of potentially large advantages for examinees involved, TC poses a serious threat to the validity of score interpretations. Hence accurately identifying individuals involved in collusion is critical as it will allow the following: their scores may be removed from the dataset so that psychometric analyses are not affected by TC; their scores may be reviewed and invalidated; compromised items may be identified and removed from the item bank or from test scores. An important special case of TC is item preknowledge, in which some candidates have obtained information about certain test questions in advance of sitting for the exam. Detecting examinees involved in TC is a challenging problem, which is due to the following:

- Examinees colluding on a particular subset of items (i.e., *compromised subset*) form a *group*, where number of such groups, their examinees, their compromised subsets, and

[1] Law School Admission Council, Newtown, PA, USA
[2] University of Wisconsin–Madison, USA

Corresponding Author:
Dmitry I. Belov, Law School Admission Council, 662 Penn Street, Newtown, PA 18940, USA.
Email: dbelov@lsac.org

their relations (which groups intersect, which compromised subsets intersect, etc.) are all unknown.

- Examinees from a *group* can be at different rooms, test centers, countries, and so on..

To make this problem tractable we introduce the following assumption (hereafter referred to as assumption A1): there are multiple nonintersecting groups each with a unique compromised subset, where compromised subsets from different groups may intersect (however, these intersections are much smaller than the subsets). The objective is to develop a statistical method to identify these groups (as opposed to a popular approach of detecting pairs of examinees who might be involved in TC).

Due to definition of TC, examinees within each group should have an unusually high response similarity on corresponding compromised subset. The *response similarity* is a collective term which may include the following: similar answer choices, similar unusually short response times, or similar answer changes. Since compromised subsets are unknown, the response similarity between two given examinees can be estimated by computing a *response similarity index* (RSI) measuring similarity between their *response vectors* (i.e., responses to the whole test). Wollack and Maynes (2017) compute RSI for each pair of examinees and cluster examinees such that two examinees having RSI value above a predetermined critical value are assigned to the same cluster; thus, detecting groups. Being involved in TC creates connections between examinees within their group, where the strength of each connection can be measured by a RSI. A natural way of analyzing connections between entities is via graph theory (Bollobas, 1998). This paper employs graph theory to analyze the clustering approach by Wollack and Maynes, finds a practical limitation with that approach, and then introduces a novel method to detecting groups.

This paper uses the ω index (Wollack, 1997) because Romero et al. (2015) demonstrated an optimality of this RSI for detecting answer copying, though the TC detector developed here can be used with any RSI. The ω index is asymmetric, meaning ω index is conditioned on estimated ability level (MAP estimate with uniform prior is used in this paper) of one of the examinees. Therefore, for each pair of examinees the RSI was computed twice, once in each direction. The RSI was assumed to be significant when at least one ω index out of the two was significant (this may cause a multiple comparison problem, which is discussed in the summary).

Data from a real credentialing program experiencing a security breach were used in this study. Data for 1,636 and 1,644 examinees (Dataset 1 and Dataset 2, respectively) were available for each of two forms of the test (Form 1 and Form 2, respectively), each with 170 operational items. The program flagged 46 examinees and 64 items with Form 1 and 48 examinees and 61 items with Form 2. It is important to recognize that the program flagged examinees and items using a variety of statistical methods and investigative techniques, and was actively searching for different types of test taking aberrancy. Therefore, in both test forms, it is not necessarily the case that all of the flagged examinees and items were detected statistically by the program, and not all relate to TC. More details on the credentialing dataset can be found in Cizek and Wollack (2017). Using these same datasets, Wollack and Maynes (2017) demonstrated that the largest clusters of examinees with significant response matching include multiple flagged examinees from the same test centers and schools. This most likely indicates that TC was present in both datasets. Therefore, we begin by creating Dataset 1* and Dataset 2* by removing all examinees flagged by the program from Datasets 1 and 2, respectively. Throughout this paper, Dataset 1* and Dataset 2* will be assumed to be free of TC, and for purposes of the simulation study, will serve as the null data into which different groups with varying magnitudes of TC will be introduced. This real-data simulation is preferred to a fully

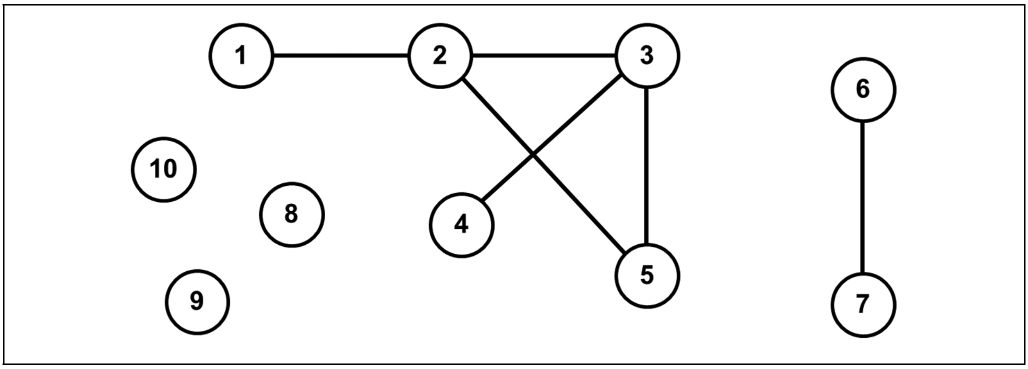

Figure 1. Illustration of a graph for 10 hypothetical examinees.

model-based simulation in which the null data are also simulated because real test data often violate major assumptions of IRT (e.g., local independence, perfect model fit, etc.).

The paper is organized as follows. First, we will interpret TC from the graph theory standpoint. Next, using the language of graph theory, we will describe a new TC detector and compare it with the Wollack and Maynes (2017) approach. Further analyses were conducted using the code written by the authors in standard C++. The source code is available upon email request.

Representing Response Data as an Undirected Graph

Commonly, a graph is defined as a set of vertices connected by edges (Bollobas, 1998), where the graph can be *directed* (vertices are connected by directed edges) or *undirected* (vertices are connected by undirected edges). In the context of TC, each vertex represents an examinee. To build edges, one starts by choosing an RSI and a significance level, α; then if for two examinees the RSI is significant, an undirected edge between the corresponding vertices is created. We denote the *undirected graph* (referred further as just the *graph*) built for the significance level α as $G(\alpha)$.

Figure 1 provides an example of the graph built from responses for 10 hypothetical examinees. The RSI is computed between all possible pairs; however, an edge is only drawn to connect those examinees for whom the RSI was statistically significant. Therefore, in Figure 1, we can see that the RSI between examinees 1 and 10 was not significant; on the other hand, the RSI between examinees 1 and 2 was significant.

The graph in Figure 1 is small and can be studied directly. With large datasets, instead of visualizing the graph directly, it is more feasible to study properties of the graph. Let us illustrate one basic property using the real datasets presented above. The sparseness (Bollobas, 1998), s, of a graph is computed as follows:

$$s = 1 - \frac{2m}{n(n-1)}, \tag{1}$$

where n is number of vertices (number of examinees) and m is number of edges. The sparseness falls within the range [0, 1], where 0 corresponds to a graph with an edge between every pair of vertices and 1 corresponds to a graph with no edges. For example, the sparseness of the graph in Figure 1 is 0.87. Figure 2 presents the sparseness for Datasets 1, 2, 1*, and 2* as a function of

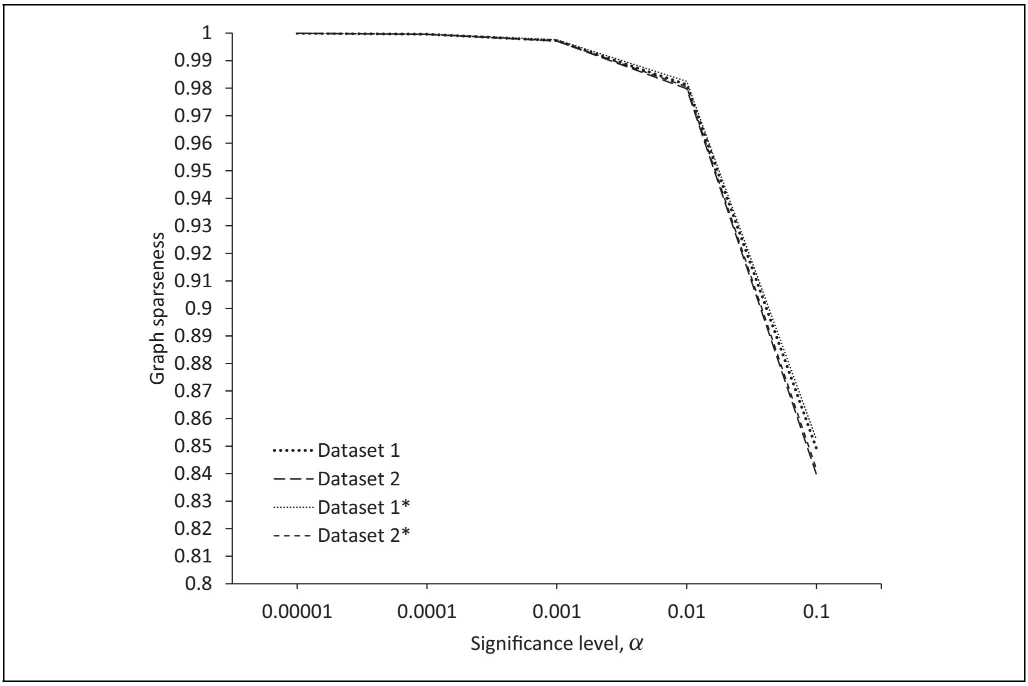

Figure 2. Association between the sparseness and α for 4 datasets.
Note. Note that the horizontal axis (log scale is used) intersects the vertical axis at 0.8.

the significance level for the RSI (ω index) used to build the graph. One can see that sparseness varies from a minimum of 0.84 (n = 1,644, m = 216,541) to a maximum of 1.0 (n = 1,636, m = 6), depending on the particular dataset and α used (but mostly depending on α). As expected, the graph $G(\alpha)$ becomes less sparse as α increases and the number of significant RSI values increases (Figure 2). It is interesting to note that in datasets such as these which include relatively few contaminated examinees—as tends to be the case in practice—sparseness does not appear to be strongly affected by test cheating, as the curves for each real dataset and its purified null counterpart are highly similar.

Detecting Groups of Examinees Involved in TC

A subgraph (Bollobas, 1998) is defined as a graph whose vertices and edges are subsets of another graph. For example, for the graph in Figure 1, the subgraph including vertices {5, 6, 7}[1] includes an edge between 6 and 7 and no edges between 5 and either 6 or 7.

In terms of graph theory, the method by Wollack and Maynes (2017), hereafter referred to as WM, finds largest connected components in the graph.[2] A *connected component* is a subgraph where any two vertices are connected by a *path* (a sequence of vertices connected by edges). For example, in Figure 1, subgraph {1, 2, 5} is a connected component, but subgraph {1, 2, 4} is not because there is no path in that subgraph from 4 to either 1 or 2. Wollack and Maynes (2017) reported only connected components larger than 4 (this cut point was defined based on an empirical analysis of all connected components, see page 144 in Wollack & Maynes). Such cut point selection is not generalizable; therefore, we modified the WM as follows. Find components sequentially (from the largest to the smallest) and remove them from the graph until the number of detected examinees (sum of sizes of found components) divided by total number of examinees becomes higher than α (to control the Type I error). Of all the connected components in Figure 1, the largest is $\{1, 2, 3, 4, 5\}$, where one can immediately see a potential issue with WM. Examinees 1 and 4 have significant RSI with only one other member of $\{1, 2, 3, 4, 5\}$. Hence, it is not clear whether these individuals are actually part of the group. In practice, this obviously may inflate the Type I error. Wollack and Maynes recognized this as a limitation and used a special adjustment for RSI $\alpha/((n-1)/2)$ to balance between the power and the Type I error. Even still, they observed that their method is greedy and tends to include examinees with few connections to the other examinees in the group. The WM is not fully statistical; it does not provide a probability of a reported component being falsely detected.

For purposes of detecting TC, one would be interested in identifying subgraphs with near zero sparseness, since they correspond to groups of examinees whose response vectors are statistically similar to one another. This makes sense from a practical standpoint. For example, if a group of examinees has preknowledge to certain subset of items then one would expect them to have similar responses on items from this subset; in the language of graph theory, for some value of α, they all should be connected in $G(\alpha)$. A subgraph in which each vertex is connected to all other vertices is called a *clique* (Bollobas, 1998); in Figure 1, subgraph $\{2, 3, 5\}$ is a clique but subgraph $\{5, 6, 7\}$ is not a clique.

If all response vectors are independent then $G(\alpha)$ is a random graph (Erdös & Rényi, 1959), where each edge occurs with probability α independently of all other edges. In the presence of TC, response vectors of examinees within each group are no longer independent. Hence, the independence between edges (connecting colluding examinees) breaks down and cliques corresponding to each group should form in the graph. How can one establish a probability that a given clique is not formed by just a chance? One approach is to estimate how unusual its size is. This can be done using a null distribution of clique size. In a random graph the probability of k given vertices forming a clique is equal to $\alpha^{k(k-1)/2}$ (Bollobas & Erdös, 1976) which is, obviously, much smaller than α. However, the probability of at least one clique of size k in a random graph with n vertices is $1-(1-\alpha^{k(k-1)/2})^{\binom{n}{k}}$, where $\binom{n}{k}$ is total number of unique subsets with k vertices built out of n vertices (Bollobas & Erdös, 1976). This probability increases for a fixed k as n grows, which dictates that a method of computing the null distribution of clique size should depend on n too. Also, the independence between response vectors may weaken in real data even without TC. For example, popular distractors (incorrect answers that appear highly plausible) or certain answer strategies (e.g., selecting always C when in doubt) break the independence. Therefore, a great care has to be taken when computing the null distribution of clique size (see the next section); otherwise, the Type I error may be inflated. Thus, the approach of this paper is to detect significantly large cliques, and it will be referred to as the *clique detector*. The approach functions as follows:

Clique Detector

Step 1: For given data, RSI, and α, build a graph $G(\alpha)$.

Step 2: Compute a null distribution of clique size (using the procedure described below). Find the critical value on this null distribution corresponding to significance level β, which is the probability of falsely rejecting a null hypothesis that a given clique is formed by chance. To simplify estimations of Type I error we set $\beta = \alpha$.

Step 3: Find a clique of maximum size (*maximum clique*) in $G(\alpha)$. If the found clique has size equal to or greater than the critical value, then report this clique, remove this clique from $G(\alpha)$ (since assumption A1 states that groups do not intersect each other), and repeat Step 3; otherwise, stop.

The above procedure detects cliques that are significantly large (due to Steps 2 and 3) and do not intersect each other (due to Step 3). From the above, it follows that each detected clique should correspond to a group of examinees that were colluding on a specific (to that group) subset of items. Clique detector (CD) resolves two major issues of the WM mentioned above: (a) the greediness of the WM method to include vertices that are potential false positives, and (b) the lack of a statistical indication of the false positive rate for detected components. The CD is a fully statistical method which explicitly and strictly controls the Type I error rate. Therefore, it is expected to perform better than WM.

At a conceptual level, the steps involved in CD are straightforward and are consistent with how one conducts normal hypothesis testing when the test statistic does not follow a known statistical distribution. However, the processes by which the maximum clique is found and the null distribution of clique size is derived are both non-trivial.

Finding the maximum clique (see Step 3 of CD above) is a well-known combinatorial optimization problem (Garey & Johnson, 1979). In general, finding the maximum clique is computationally hard; however, for a sparse graph it can be found quickly, such as with the branch-and-bound algorithm by Wood (1997) applied in this paper. As was demonstrated in the previous section, the sparseness of the graphs for all four real datasets is above 0.99 for α at or below 0.001 (see Figure 2). Hence, further computational studies were performed using $\alpha = 0.001$.

The null distribution of clique size can be computed from a large sample of null cliques, where null cliques are drawn from a sample of null graphs. One may think of this process as follows: first, build a sample of null graphs; second, draw multiple random cliques (sampling with replacement) from each graph in the sample and then compute the histogram of clique size.

Methods to Build a Null Graph

There is no single, agreed-upon method by which to create the null graph. We are going to propose three different methods to generate the null graph, apply each of those methods to the empirical datasets, and evaluate the results for purposes of determining which method produces the best properties.

The first method (Method 1) to generate the null graph uses item parameter estimates for an IRT model (obtained from either the current or prior test administration) and a null latent trait distribution to generate null responses for the same number of examinees as in the original dataset. Using the simulated dataset, for a specified α, RSIs are computed between all possible examinee pairs and the null graph is built. With respect to the original graph $G(\alpha)$ built for a given dataset, Method 1 guarantees to preserve only the number of vertices.

The second method (Method 2) is based on the theory of random graphs developed by Erdös and Rényi (1959). The method views α as the probability of an edge being randomly formed between any two vertices. Hence, a null graph can be formed by pairing each vertex with every other vertex and randomly creating an edge between them with probability α. However, creating the null graph in this manner will likely result in a graph with a different number of edges than in the original graph $G(\alpha)$ built for a given dataset, which may affect the distribution of clique size. Therefore, in Method 2, an iterative approach is used in which a random pair of vertices is selected and an edge is created between them with probability α until the total number of edges created matches the number from $G(\alpha)$. With respect to $G(\alpha)$, Method 2 guarantees to preserve both the number of vertices and the number of edges.

Before diving into details of the third method (Method 3), one natural constraint on the null distribution of clique size has to be pointed out. Given the data to be analyzed, RSI, and α, consider the corresponding graph $G(\alpha)$. Cliques of size 1 (isolated vertices in $G(\alpha)$) correspond to examinees that are not connected to any other examinees, and their proportion is specific to the data, RSI, and α. That proportion establishes a constraint on the Type I error and the power when detecting pairs of examinees with significant RSIs. It is natural to preserve this constraint when detecting significantly large cliques. In other words, the proportion of cliques of size 1 in the null graph should match their proportion in $G(\alpha)$; otherwise, the null distribution of clique size computed from null graphs may produce incorrect critical values, which will inflate the Type I error or reduce the power of the clique detector. One method to satisfy this constraint is to apply multiple modifications to $G(\alpha)$ such that each modification preserves *vertex degrees*. The vertex degree for a particular vertex refers to the number of other vertices connected to it (Bollobas, 1998). Vertex degrees represent inherent dependencies in the data that are difficult to model directly; in the context of this paper, the particular answers selected by each examinee could result in connecting him or her to other examinees, thereby affecting the vertex degrees. However, it is important to note that the majority of these dependencies between examinees do not relate to TC: for example, an examinee being connected to 10 other examinees does not necessarily mean that these 10 examinees are all interconnected.

The null graph by Method 3 results from a Markov chain applied to $G(\alpha)$, where each iteration performs a random double edge swap to preserve the vertex degrees in the graph (Fosdick et al., 2018). In a random double edge swap, at each iteration of the Markov chain, two edges from $G(\alpha)$ are randomly selected and replaced with two new edges in such a way as to maintain the vertex degrees for all vertices. This iterative process continues for an extended time until the systematic relationship that existed due to potential non-null effects is eliminated (burn-in), after which, successive swaps are assumed to produce null graphs. In other words, if a graph has cliques that are larger than expected by chance then after some burn-in period these cliques should be eliminated. Figure 3 illustrates the result of one double edge swap applied to the graph from Figure 1. The new graph has vertex degrees exactly the same as the original graph (although note that the maximum clique size in the new graph is 2, not 3).

Once a sample of null graphs is created (using Methods 1, 2, or 3), building the null distribution of clique size requires identifying multiple random cliques and recording their size. In this study, the heuristic CLIQUE described by Wood (1997) was modified to find random cliques in a graph by selecting each vertex for a clique randomly (see details in Appendix).

Figure 4A shows probability mass function (PMF) of clique size estimated from graph $G(\alpha)$ built for Dataset 1* (with number of vertices 1590 and number of edges 3065), where $\alpha = 0.001$. Figures 4B–D show PMFs of clique size estimated from null graphs built by Methods 1–3, respectively. Each PMF was estimated from a sample of 1000 random cliques drawn from the corresponding graph. One can see a large deviation between the distribution in Figure 4A and those in 4B and 4C. In particular, Method 1 appears to produce an overly simplistic null graph (with number of vertices 1590 and number of edges 565) that includes an over-abundance of cliques of size 1 and virtually no cliques larger than pairs of examinees. Method 2 also produces an overly simplified null graph (with number of vertices 1590 and number of edges 3065), though its characteristics are rather different than those from the Method 1 graph. Method 2 appears to do a poor job of capturing the natural variability in clique size that exists in real data, with approximately 98% of its values equaling 2. Importantly, Methods 1 and 2 both appear to do poorly at estimating the right tail of the distribution,

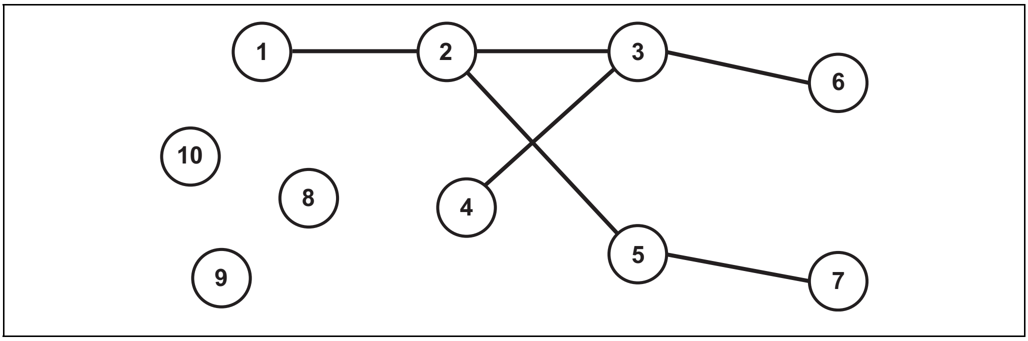

Figure 3. Transformed graph from Figure 1 after one double edge swap was applied.
Note. This new graph has vertex degrees exactly the same as the original graph in Figure 1.

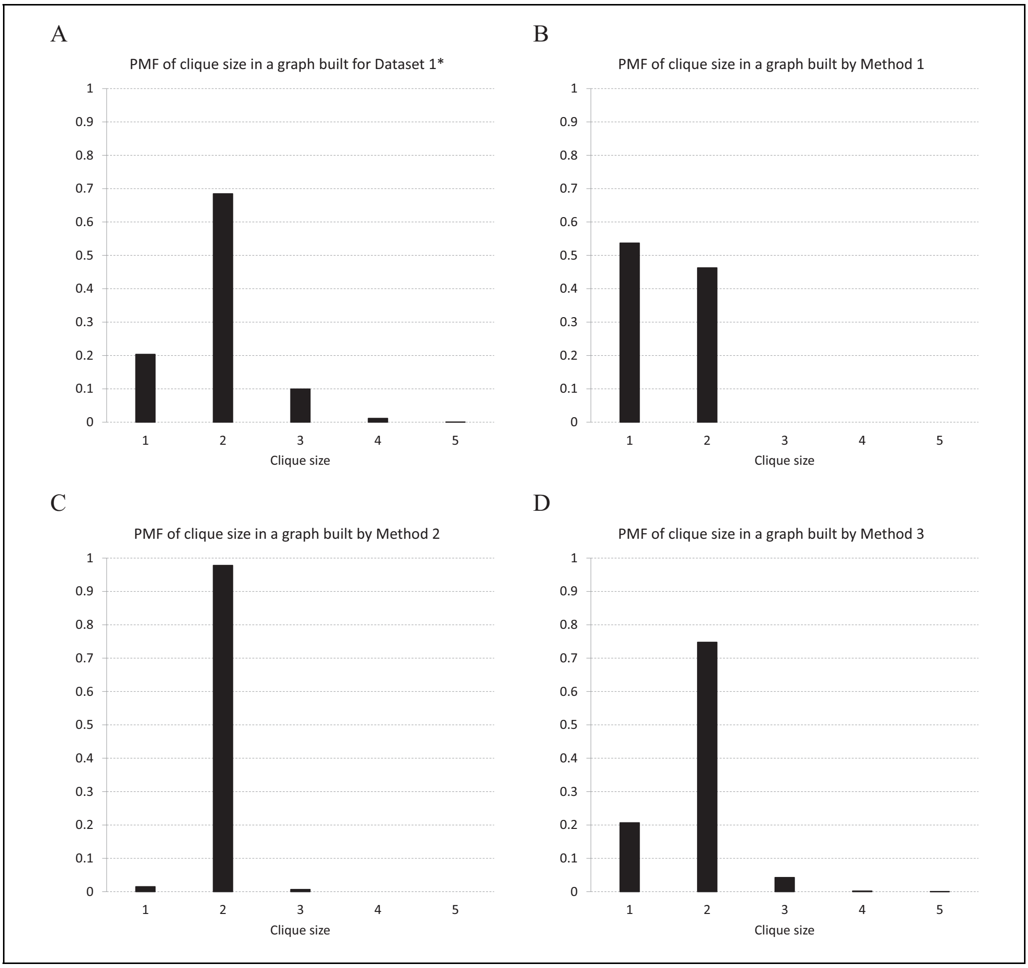

Figure 4. Four distributions of clique size. (A) PMF of clique size in a graph built for Dataset 1*, (B) PMF of clique size in a graph built by Method 1, (C) PMF of clique size in a graph built by Method 2, and (D) PMF of clique size in a graph built by Method 3.
Note. PMF = probability mass function.

suggesting that the critical values of clique size they provide will be too low and will result in an inflated Type I error.

In contrast, the PMF of clique size in a graph built by Method 3 (with burn-in period of 5,000 double edge swaps applied to $G(\alpha)$) shown in Figure 4D is considerably more similar to the PMF shown in Figure 4A. Method 3 appears to best capture the true variability in the null distribution, and also appears to reproduce the right tail more accurately than other methods. Similar results (available from authors upon request) were found for Dataset 2*. Based on the above analysis, the null distribution of clique size is computed as follows:

MCMC Procedure

Step 1: For given data, RSI, and α, build a graph $G(\alpha)$.
Step 2: Perform a burn-in period, where 10,000 double edge swaps are applied to $G(\alpha)$.
Step 3: Apply another 10,000 double edge swaps, where the PMF of clique size is updated after each 100 double edge swaps by sampling (with replacement) 1,000 random cliques from the current graph.

Comparison Study

This section will analyze advantages and limitations of CD via comparison with WM, where significance level α was set to 0.001. Whereas Wollack and Maynes used the *M4* index (Maynes, 2014) as their RSI, in this paper, the ω index (Wollack, 1997) is used to facilitate a more direct comparison with the CD. We checked our implementation of the WM on the same data for the same $\alpha = 0.001$ (and the correction for RSI $\alpha/((n-1)/2)$) and found a strong agreement with the results reported by Wollack and Maynes. In particular, for Dataset 1 we found that the largest connected component has 10 examinees, where 9 of them were also reported in Table 6.4 (see cluster #1 with 12 examinees on p. 146 in Wollack & Maynes); similarly, for Dataset 2 the majority of examinees in the largest connected component were also reported in that table.

TC can take many forms, but for purposes of this study, we simulated it as multiple examinees copying answers from a common source. Groups were simulated by manipulating the percentage of copied items (30%, 50%, or 70%), total number of colluding examinees (20, 100, or 200), and group size (5, 10, or 20). As mentioned in the Introduction, the simulated TC was added to either Dataset 1* or Dataset 2*. Each scenario of TC was simulated as follows: examinees were randomly selected from given data (Dataset 1* or Dataset 2*) and then cloned (i.e., their response vectors were cloned); these clones were randomly partitioned into groups; for each group, the specific number of compromised items were randomly selected and then the answers of a randomly chosen clone to these items were shared with other clones in the group. The number of groups of each size varied as a function of the total number of colluding examinees, as shown in Table 1. Thus, there were 9 simulated scenarios of TC and each scenario had different number of groups of different sizes.

Five outcome measures were estimated. The Type I error was defined as follows:

$$\frac{[\text{number of detected non} - \text{colluders}]}{[\text{number of non} - \text{colluders}]} \qquad (2)$$

The power was computed according to the following:

$$\frac{[\text{number of detected colluders}]}{[\text{number of colluders}]} \tag{3}$$

The precision was computed according to the following:

$$\frac{[\text{number of detected colluders}]}{[\text{number of detected examinees}]} \tag{4}$$

The next two characteristics evaluate performance of detecting group structure. Wollack and Maynes (2017) recommended that the quality of group-level detection be evaluated through consideration of both group power and group integrity. For purposes of this study, group power is computed as follows:

$$\frac{[\text{number of detected groups}]}{[\text{number of groups}]}, \tag{5}$$

where group is considered detected if at least one examinee from that group was detected. Although Wollack and Maynes evaluated group integrity through visual inspection, we propose using the group precision as a quantitative measure of group integrity:

$$\frac{1}{k}\sum_{i=1}^{k}\frac{h_i}{s_i}, \tag{6}$$

where k is number of detected groups (*cliques* for CD and *connected components* for WM), s_i is size of detected group i, and h_i is the largest number of detected colluders in detected group i that belong to the same group. Group precision is intended to provide a measure of the extent to which the individuals detected all belong to the same group.

Means of the above characteristics are presented in Tables 2 to 6 where each mean was estimated from 10 repetitions per each simulated TC scenario added to Dataset 1*. One can see that in most cases the CD substantially outperforms the WM. Similar results (available from authors upon request) were found when simulated TC scenarios were added to Dataset 2*.

Summary

This paper introduces a novel methodology for detecting groups of examinees involved in TC based on graph theory. Graphs provide an ideal structure to describe relations among examinees; TC is but one potential application. Due to the nature of test cheating, multiple problems and methods in test security can be reduced to problems on graphs and graph algorithms, respectively. For example, we showed that the recently developed TC detector by Wollack and Maynes (2017) is equivalent to finding the largest connected components in a graph. In our opinion, rich apparatus developed by the discrete math community for abstract graphs can find multiple applications in test security.

In this paper, detecting groups of examinees involved in TC was reduced to detecting significantly large cliques in a graph built for given data, RSI, and α. The comparison study indicated that the clique detector outperforms the detector by Wollack and Maynes (2017) on all the following measures of performance: Type I error, power, precision, group power, and group precision (see Tables 2 to 6). These results were expected as we introduced CD as a way to fix issues with WM. Tables 3 and 5 show that the power of CD slightly decreases as the number of colluding examinees increases, and the power decreases slightly as the percentage of compromised items changes from 50% to 70%. This was also expected since we did not control for the

Table 1. Number of Groups of Different Sizes.

Total number of colluding examinees	Number of groups of size 5	Number of groups of size 10	Number of groups of size 20
20	2	1	0
100	10	3	1
200	10	5	5

Table 2. Type I Error by WM and CD.

	Percentage of compromised items					
	30%		50%		70%	
Number of colluding examinees	WM	CD	WM	CD	WM	CD
20	0.005	0.007	0.005	0.006	0.002	0.006
100	0.012	0.008	0.012	0.007	0.006	0.007
200	0.013	0.006	0.013	0.005	0.008	0.005

Note. WM = Wollack and Maynes; CD = clique detector.

Table 3. Power by WM and CD.

	Percentage of compromised items					
	30%		50%		70%	
Number of colluding examinees	WM	CD	WM	CD	WM	CD
20	0.36	0.72	0.50	1.00	0.50	1.00
100	0.18	0.65	0.21	0.98	0.21	0.94
200	0.10	0.70	0.11	0.94	0.10	0.91

Note. WM = Wollack and Maynes; CD = clique detector.

intersection between compromised subsets (see the assumption A1 formulated above); in particular, intersections between compromised subsets grew with the percentage of compromised items and the number of groups, thus, violating the assumption A1.

Tables 5 and 6 indicate that CD greatly improves the detection of groups compared to WM. This is the major advantage of the clique detector. The CD identifies groups of examinees involved in TC without using any knowledge about compromised subsets—parts of test that were affected by TC. After each clique is detected, those parts can be reconstructed as follows: the items with common responses between all examinees from a detected clique can be used as an approximation of a corresponding compromised subset (see more details in Pan & Wollack, 2020). As a result, practitioners know which items should be replaced, revised, or retired; thus, negating a potential harm by TC. Also, test scores may be adjusted by removing the compromised items, or they may be invalidated by detecting users of the compromised items using the method described in Belov (2016, 2017) or Wang et al. (2019).

Table 4. Precision by WM and CD.

	Percentage of compromised items					
	30%		50%		70%	
Number of colluding examinees	WM	CD	WM	CD	WM	CD
20	0.48	0.57	0.54	0.67	0.74	0.66
100	0.50	0.85	0.53	0.90	0.69	0.89
200	0.50	0.93	0.51	0.96	0.63	0.96

Note. WM = Wollack and Maynes; CD = clique detector.

Table 5. Group Power by WM and CD.

	Percentage of compromised items					
	30%		50%		70%	
Number of colluding examinees	WM	CD	WM	CD	WM	CD
20	0.33	0.77	0.33	1.00	0.33	1.00
100	0.08	0.66	0.08	0.99	0.08	0.94
200	0.06	0.72	0.07	0.90	0.05	0.82

Note. WM = Wollack and Maynes; CD = clique detector.

Table 6. Group Precision by WM and CD.

	Percentage of compromised items					
	30%		50%		70%	
Number of colluding examinees	WM	CD	WM	CD	WM	CD
20	0.48	0.53	0.54	0.60	0.74	0.59
100	0.48	0.79	0.51	0.87	0.68	0.85
200	0.49	0.89	0.47	0.91	0.63	0.91

Note. WM = Wollack and Maynes; CD = clique detector.

Table 2 indicates an inflation of the Type I error for both WM and CD, which can be explained by computing the ω index twice for each pair of examinees and by multiple subgraphs (connected components for WM and cliques for CD) being statistically tested. Another reason is that Dataset 1* (and Dataset 2*) could still contain colluding examinees and the Type I error rates may have been influenced by that. In any case, to remove the effect of multiple comparisons, one may use a simple correction like $\alpha/2$ or use a symmetric RSI, like *M4* (Maynes, 2014) or the generalized binomial test (van der Linden & Sotaridona, 2006).

If the same α is used then WM and CD can be viewed as two extremes. WM detects connected components (although WM does not apply a hypothesis test on the group level), which may have a higher power but potentially introduce a higher rate of false positives. On the other hand, CD detects fully connected subgraphs (i.e., cliques). Therefore, this approach may sacrifice the power, as there may be a vertex connected to most of the vertices in a detected clique but, obviously, this vertex will not be detected since it is not connected to all vertices in the clique. To address this issue one may search for a near-clique—a subgraph where every vertex is connected to most of vertices in the subgraph. CD can be easily adapted to support near-cliques by using the following procedure:

Extending a Clique to a Near-Clique

Step 1: A vertex is randomly drawn (sampling without replacement) from a list of vertices not included in the clique.

Step 2: If the vertex is connected to almost all members of the clique (e.g., at least 90% of them) then the vertex is added to the clique.

Step 3: If the list is not empty then return to Step 1, otherwise stop.

This procedure always converges because each call of Step 1 removes one vertex from the finite list of vertices. However, this heuristic does not guarantee to return a near-clique of maximum size; for more information on this topic and for more advanced methods see, for example, Jain and Seshadhri (2020).

Other extensions of the presented methodology are possible. First, RSI can be extended by using response time data; for example, a similar unusually quick response may be an indication of item preknowledge. Second, studying how the distribution of clique size in $G(\alpha)$ changes with α may provide some visual clues on potential TC presence in a given dataset. Third, a more general approach is to operate on a weighted graph, where all vertices are connected with each other and each edge has a weight equal to the corresponding value of RSI.

Based on the presented research, several recommendations for a practitioner can be provided. For large datasets with low expected levels of TC (a common situation considered here), we recommend using MCMC to generate null graphs (see Method 3). In other situations, one may consider using several methods, for example, start with Method 3 and then apply more aggressive Methods 1–2. It might also be reasonable to look for curious demographic patterns among the detected examinees (e.g., many from a common test center, country, graduates of a common institution, etc.) for purposes of informing the investigative process and potentially uncovering leads to other individuals involved in collusion who were not detected. For example, after identifying these patterns, all examinees can be partitioned based on corresponding characteristics (common test center, country, graduates of a common institution, etc.), then the detector is applied for each partition separately with larger α to increase the power.

If two significantly large cliques are intersecting then two corresponding groups have intersecting compromised subsets. Larger intersection between the two compromised subsets violates the assumption A1 (formulated in the Introduction) and, therefore, causes a larger intersection between the two cliques. In that case, removal of one clique from the graph may reduce the second clique below the critical value, thus, reducing the power of CD (see Tables 3 and 5). To address this issue, one may sample random cliques from the graph (without removing them) and report cliques of a significant size (this is not covered in this paper; however, in this case one should be concerned about the multiple comparison problem). Two additional points on the assumption A1 can be made. First, if multiple different groups have the same compromised subset then from the test security standpoint they can be interpreted as a single group. Thus, the assumption about multiple groups each with a unique compromised subset is general. Second, the assumption about nonintersecting groups is also general. This follows from an observation that for any two groups A and B with compromised subsets C and D, respectively, they can be partitioned into three nonintersecting groups: $A \backslash B$ with compromised subset C, $A \cap B$ with compromised subset $C \cup D$, and $B \backslash A$ with compromised subset D. However, if $A \cap B$ is not empty then the intersection between $C \cup D$ and D is D (the intersection between compromised subsets is not small), which violates the assumption A1.

Another limitation is that the suggested approach is computationally fairly demanding. We addressed this by using low $\alpha = 0.001$ (producing sparse graphs) and programming all analyses using C++, which helped to compute all results from Tables 2 to 6 in about 100 minutes on regular PC.[3] However, the same PC performing the same analysis running in R would likely have taken many hours.

Appendix

The process by which a random clique is found is as follows (this is a trivial modification of the heuristic CLIQUE described by Wood (1997):

```
Step 1. Create an empty subset of vertices
Step 2. Select a random vertex $x$ from the graph and add it to the subset
Step 3. Create a list with all vertices that are connected to the vertex $x$
Step 4. While the list is not empty repeat the following:
    Step 4.1. A vertex $y$ is randomly drawn (sampling without replacement) from the list
    Step 4.2. Add the vertex $y$ to the subset
    Step 4.3. Remove from the list all vertices that are not connected to $y$.
```

Because of Step 3 and Step 4.3, the above procedure builds a subset of vertices that are all connected in the graph (i.e., a clique). Let us illustrate above steps of finding a random clique using the graph in Figure 1. Assuming that at Step 2 above the vertex $x = 2$ is selected, the following steps illustrate changes in the list and in the outcome:

Step 2. Subset={2}
Step 3. List={1, 3, 5}
Step 4. List is not empty
Step 4.1. Vertex 1 is drawn from the List resulting in List={3, 5}
Step 4.2. Subset={2, 1}
Step 4.3. List becomes empty (3 and 5 are removed because 1 is not connected to 3 and 5) and the resultant clique is {1, 2}. Note, if 3 or 5 were drawn on Step 4.1 then the resultant clique would be {2, 3, 5}.

Acknowledgment

The authors thank the Editor, John R. Donoghue, the Associate Editor, Yi Zheng, and three anonymous reviewers for their valuable comments and suggestions on previous versions of the manuscript.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

ORCID iD

Dmitry I. Belov https://orcid.org/0000-0001-7800-8959

Notes

1. For simplicity, we will refer to a subgraph by just enumerating its vertices without enumerating its edges; hence, this subgraph can be referred as {5, 6, 7}.
2. Wollack and Maynes (2017) technically used the nearest neighbor clustering method; in graph theory terms, nearest neighbor clustering is equivalent to finding connected components.
3. Intel Core i7-4770 CPU 3.40GHz, 8GB RAM, 64-bit Microsoft Windows 7.

References

Belov, D. I. (2016). Comparing the performance of eight item preknowledge detection statistics. *Applied Psychological Measurement*, *40*, 83–97.

Belov, D. I. (2017). On the optimality of the detection of examinees with aberrant answer changes. *Applied Psychological Measurement*, *41*, 338–352.

Bollobas, B. (1998). *Modern graph theory*. Springer.

Bollobas, B., & Erdös, P. (1976). Cliques in random graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, *80*(3), 419–427.

Cizek, G. J., & Wollack, J. A. (2017). Exploring cheating on tests. In G. J. Cizek & J. A. Wollack (Eds.), *Handbook of quantitative methods for detecting cheating on tests* (pp. 3–19). Routledge.

Erdös, P., & Rényi, A. (1959). On random graphs I. *Publicationes Mathematicae Debrecen*, *6*, 290–297.

Fosdick, B. K., Larremore, D. B., Nishimura, J., & Ugander, J. (2018). Configuring random graph models with fixed degree sequences. *SIAM Review*, *60*(2), 315–355.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. W.H. Freeman.

Jain, S., & Seshadhri, C. (2020). Provably and efficiently approximating near-cliques using the Turán shadow: PEANUTS. In *Proceedings of the web conference 2020 (WWW '20)* (pp. 1966–1976). Association for Computing Machinery. https://doi.org/10.1145/3366423.3380264

Maynes, D. D. (2014). Detection of non-independent test taking by similarity analysis. In N. M. Kingston & A. K. Clark (Eds.), *Test Fraud: Statistical detection and methodology* (pp. 53–82). Routledge.

Pan, Y., & Wollack, J. A. (2020, April). *An iterative unsupervised-learning-based approach for detecting item preknowledge* [Paper presentation]. Annual Meeting of the National Council on Measurement in Education, San Francisco, CA, United States.

Romero, M., Riascos, Á., & Jara, D. (2015). On the optimality of answer-copying indices: Theory and practice. *Journal of Educational and Behavioral Statistics*, *40*, 435–453.

van der Linden, W. J., & Sotaridona, L. (2006). Detecting answer copying when the regular response process follows a known response model. *Journal of Educational and Behavioral Statistics*, *31*, 283–304.

Wang, X., Liu, Y., Robin, F., & Guo, H. (2019). A comparison of methods for detecting examinee preknowledge of items. *International Journal of Testing*, 19(3), 207–226.

Wollack, J. A. (1997). A nominal response model approach for detecting answer copying. *Applied Psychological Measurement*, *21*, 307–320.

Wollack, J. A., & Maynes, D. (2017). Detection of test collusion using cluster analysis. In G. J. Cizek & J. A. Wollack (Eds.), *Handbook of quantitative methods for detecting cheating on tests* (pp. 124–150). Routledge.

Wood, D. R. (1997). An algorithm for finding a maximum clique in a graph. *Operations Research Letters*, *21*, 211–217.