

Random Walk in Graphs Analyzing and Applications

Yunqing Cai *

Nanjing University of Aeronautics and Astronautics

* Corresponding Author Email: 3300383832@qq.com

Abstract. As a fundamental topic in probability theory, the random walk has a wide range of applications in real-world scenarios. It is widely employed not only in financial and economic fields such as stock price forecasting and video game design, but also in more abstract domains including graph analysis, especially network graph analysis. This study demonstrates that the random walk is effective in dealing with network graph-related problems. During the experience, MATLAB is used to test the ability of random walk in identifying the difference between two graphs, and it shows that random walk shows some advantage in identifying difference between network graphs compared to spectral matching, which has been viewed as authentic method relating to graphs for many years. What's more, it also shows possibility that when combined with big data models, random walk can further show great potential in investigating the relationships among two or more network graphs.

Keywords: Random Walk; Graph Analysis; Graph Comparison; Structural Change Detection; PageRank; Spectral Matching.

1. Introduction

1.1. Background of Random Walk

Imagine a drunker who moves from a fixed place, and he totally randomly walks one step per second, only left or right direction. The walking process of the drunker is called random walk, and it can be showed like the following:

$$X_1 = \begin{cases} +1 & \text{with probability } p \\ -1 & \text{with probability } (1 - p) \end{cases} \quad (1)$$

And after n steps, if viewing the position of the fixed place as the origin O , the position of the drunker $S_i = \sum_{j=1}^i X_j, n \geq 1$. It is firstly promoted by Karl Pearson in 1905, and then showed great potential in all aspects.

Furthermore, Pólya (1921) proved that a random walk on certain lattices is recurrent, meaning that the walker will return to the starting point with probability one given an infinite number of steps.

1.2. Applications of random walk

Random walk serves as a powerful mathematical tool for addressing various challenging problems in many prospections from algebra, statistics and physics to economy, biologic and game design.

1. Random Walk in Physics

Random walk holds considerable significance within the field of physics, particularly in thermology. According to the theory of Brownian motion proposed by Robert Brown in 1872, microscopic particles exhibit continuous disordered motion whose velocity varies with temperature; such behavior can be effectively modelled using random walk principles.

Beyond Brownian motion, random walk methodologies also support the predictive analysis of pollutant diffusion in atmospheric and marine environments. Since particle movement characteristics are closely correlated with temperature variations, random walk can further facilitate heat transfer prediction. Additionally, the random walk particle tracking approach is widely adopted to investigate and characterize temperature distribution patterns in physical systems.

2. Random Walk in Algebra and Statistics

Beyond physical domains, random walk exhibits extensive applicability in algebra and statistics. It serves as a powerful numerical approximation tool for addressing intractable mathematical

problems, particularly stochastic differential equations (SDEs). Since SDEs incorporate both deterministic drift terms and stochastic fluctuation components, analytical solutions are unattainable for most complex cases. Random walk enables probabilistic simulation of such stochastic processes, and the corresponding mathematical expectation can be adopted to derive reliable approximate solutions. Accordingly, random walk provides a statistically feasible paradigm for obtaining numerical solutions in algebraic computation.

Furthermore, conventional uniform sampling suffers notable limitations when handling large-scale sample spaces. It often fails to efficiently characterize the inherent distribution of the original population and may introduce sampling bias. In contrast, random walk-based sampling effectively resolves this issue by conducting stochastic traversal across the entire sample space. The output proportion of sampled outcomes naturally conforms to the underlying distribution of the original population. This advantage becomes particularly prominent when the dataset contains imbalanced sample categories or dominant majority subgroups within the overall sample space.

3. Random Walk in Economy

Furthermore, random walk has demonstrated significant potential in economic and financial research. Cox (1978) proposed a seminal stock price prediction model, which formally establishes that stock price dynamics can be rigorously characterized by a random walk process. The model is expressed as follows:

$$S_n = S_0 \exp \left(\frac{\ln u - \ln d}{2} Z_n + \frac{\ln u + \ln d}{2} n \right) \quad (2)$$

When S_n is the stock price at n th time step, S_0 is the initial stock price, u and d is upload and download factors respectively, and Z_n is the position of symmetric random walk after n steps.

Accordingly, this framework confirms that stock prices can be effectively modeled and analyzed using random walk theory.

4. Random Walk in Game Designing

Beyond academic research, random walk also exhibits practical value in game development. In modern open-world games, large quantities of non-player characters (NPCs) are required to enrich the gaming environment. Traditional fixed patrol routes often lead to rigid and repetitive behaviors, which reduce player engagement. By introducing random walk mechanisms, NPC movement trajectories can be dynamically generated, creating unpredictable and natural behavioral patterns. This enhances environmental diversity, maintains long-term gameplay novelty, and effectively improves players' immersive experience within the game world.

1.3. Previous Researches, Limitations and innovation

Although random walk has demonstrated extensive applications in mathematical research, its potential in graph theory and graph data analysis remains far from fully exploited. Existing studies on random walk in graphs primarily focus on two core directions: one is leveraging random walk for model checking (Haslum, 1999) and solving classical graph problems, such as the sublinear-time triangle counting without full graph traversal (Bera & Seshadhri, 2020); the other is developing novel graph learning algorithms, such as the RAW-GNN framework, which integrates random walk with recurrent neural network aggregation to simultaneously capture structural regularities in homogeneous graphs and semantic discrepancies in heterogeneous graphs (Jin & Wang, n.d.). However, current research has not devoted sufficient attention to the in-depth exploration of random walk in graphs beyond basic traversal and reading operations, leaving significant room for further methodological innovation and application expansion.

1.3.1. Previous Researches

There are several different researches regarded to random walk and graphs, and the summery are as follow:

1. Model checking

Haslum (1999) argued that a random walk on a network graph can be treated as a Markov process over all vertices, with the expected number of steps required to return to the origin given by: $d(v)2|E|$. Where $|V|$ denotes the number of vertices and $|E|$ denotes the number of edges. Based on this result, one can introduce a small risk parameter ε and perform a random walk of length $\frac{1}{\varepsilon}|V||E|$. If the walk fails to reach the target vertex within this bound, the probability that the vertex is unreachable is greater than $1 - \varepsilon$; otherwise, the vertex is considered reachable. This framework can be generalized to various other scenarios. While it performs optimally on Eulerian graphs, further analysis confirms that it can also be applied to non-Eulerian graphs, provided that the symmetry of the graph is first verified.

A key advantage of random walk-based model checking over traditional methods is its exceptional memory efficiency: it only requires storing the current state and a step counter, whereas conventional approaches must maintain the entire state space in memory. Although the time complexity of the random walk method is (to be specified), it is inherently parallelizable, which effectively reduces the average runtime of the model checking procedure.

Furthermore, the theoretical upper bound on the required number of steps ($|V||E|$, established by Aleliunas et al., 1979) is significantly larger than the number of steps typically needed in practice. This discrepancy allows for further reduction in the average execution time and enables more extensive exploitation of parallelization potential.

2. Tetris

TETRIS is an algorithm proposed by Bera and Seshadhri for estimating the number of triangles within a single graph. The core principle of TETRIS is as follows: the algorithm first performs a random walk to collect edges along the traversal path, and then conducts secondary sampling over the collected edge set. For each sampled edge, the algorithm examines whether it can form a triangle with its neighboring vertices. A unique assignment rule based on vertex degree is further adopted to determine the edge to which each detected triangle belongs.

Empirically, TETRIS achieves remarkable estimation performance. When merely observing approximately 3% of the entire graph across 100 independent runs, the method attains a worst-case relative error of only 5% and a median relative error of roughly 2%, demonstrating high accuracy and strong robustness. Furthermore, additional tests verify that the selection of the initial seed vertex exerts negligible influence on the final estimation, which confirms the algorithm's outstanding robustness with respect to starting-point randomness.

In conclusion, TETRIS enables efficient triangle counting for large-scale complex graphs, requiring fewer sampling steps while maintaining superior accuracy and robustness compared with conventional baseline methods.

3. RAW-GNN

RAW-GNN (Jin & Wang, 2022) is a novel graph neural network method that focuses on handling both homophily and heterophily information simultaneously. Unlike traditional graph neural network methods, which are based on the homophily hypothesis of graphs and thus perform poorly in heterogeneous graphs, RAW-GNN adopts a new aggregation mechanism to process graph data. Specifically, RAW-GNN employs breadth-first random walk to capture homophily information and depth-first search to collect heterophily information, enabling it to be applicable to both homophilic and heterophilic graphs.

The framework of RAW-GNN mainly consists of four components: Random Walk Generator, RNN-based Path Aggregator, Attention-based Intra-Policy Combiner, and Inter-Policy Combiner. The Path Aggregator utilizes a recurrent neural network, such as a gated recurrent unit, to process the ordered attribute sequence of nodes along the path, which helps alleviate the issues caused by attribute differences between adjacent nodes in heterogeneous graphs. Subsequently, an attention mechanism is adopted to learn the importance of different paths under the same strategy, thereby generating strategy-specific node embeddings. Finally, the embeddings from different strategies are

concatenated to minimize information mixing, allowing the model to automatically balance between homogeneity and heterogeneity based on network characteristics.

Furthermore, the effectiveness of RAW-GNN has been validated on real-world and complex graph datasets. The results demonstrate that RAW-GNN exhibits competitive performance in the analysis of both homophilic and heterophilic graphs.

1.3.2. Limitations

Existing literature demonstrates that random walk techniques have been widely applied to various graph-related tasks. Nevertheless, current research still lacks sufficient insights and in-depth exploration into the capability of random walk for detecting structural relationships and similarities across different graphs. This research gap leaves considerable room for further investigation into cross-graph analysis and comparative evaluation based on random walk mechanisms.

1.3.3. Innovations

Therefore, this study verifies that random walk possesses considerable potential for graph analysis and cross-graph comparison, especially in scenarios requiring the evaluation of multiple graphs. Furthermore, this approach is compared with Spectral Matching, a classical and authoritative baseline widely adopted for two-graph and multi-graph matching tasks.

2. Basic theory

2.1. Random Walk in Analyzing Graphs

As a fundamental concept in probability theory, a random walk describes a stochastic process where a particle moves randomly in a given space. Its formal definition on a graph is presented as follows:

Let $G = (V, E)$ be a graph, where V denotes the set of vertices and E denotes the set of edges. Let $d(v_i)$ represent the degree of vertex V_i , i.e., the number of adjacent vertices. The transition probability from vertex V_i to vertex V_j is defined by:

$$P_{ij} = P(X_{t+1} = V_j | X_t = V_i) = \begin{cases} \frac{1}{d(v_i)}, & (V_i, V_j) \in E, \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

This indicates that, at vertex V_i , the walker moves to each neighboring vertex with equal probability.

Random walk is commonly employed to analyze large-scale graphs that are infeasible for manual visual inspection. While large graphs inherently contain richer information, they are also more challenging to explore comprehensively, often resulting in only partial or superficial insights being extracted. Random walk addresses this challenge by generating sequential representations of the graph: starting from a specified vertex, the walker traverses the graph for a fixed number of steps, producing multiple distinct node sequences. Drawing inspiration from the Word2Vec framework, these sequences are then used to train a Skip-Gram model, yielding low-dimensional embedding vectors for each vertex. Subsequent multi-faceted analysis of these embeddings enables effective graph understanding and knowledge discovery.

Furthermore, random walk is closely related to Markov chains, with the former being a special case of the latter, albeit with key distinctions. As will be elaborated in the subsequent introduction to graph traversal via random walk, the node sequences generated by this process constitute a Markov chain, justifying the classification of random walk as a type of Markovian process. However, a general Markov chain may possess an arbitrary state space lacking group structure, and its transition probabilities can exhibit non-homogeneous behavior, varying arbitrarily with the current state. (P. Haslum, 1999).

2.2. Spectral Matching

In the experiment, the random walk method is compared with Spectral Matching method. Spectral Matching is a classic graph matching algorithm that formulates node correspondence as an eigen-decomposition problem of an affinity matrix. Let G_1 and G_2 denote two graphs with n and m vertices, respectively. All candidate correspondences (i, j) between node $i \in V_1$ and node $j \in V_2$ are enumerated. An affinity matrix M of size $(mn) \times (mn)$ is constructed, where each entry $M_{(i,j),(c,d)}$ encodes the compatibility between two pairs of correspondences. Typically, this entry is assigned a value of 1 if edges (i, c) in G_1 and (j, d) in G_2 both exist; otherwise, it is set to 0. The principal eigenvector of M (corresponding to the largest eigenvalue) is then computed, and its entries serve as soft matching scores for each candidate node pair. Finally, a discrete one-to-one matching result is obtained by solving a linear assignment problem on the reshaped score matrix, commonly via the Hungarian algorithm.

3. Experiment

3.1. Plot and results

Since random walk enables graph analysis and multi-aspect information extraction from individual graphs, it also possesses the potential to analyze two distinct graphs and identify their differences.

To verify this hypothesis, an experiment was conducted using MATLAB as follows:

First, a network graph is constructed using MATLAB. To serve as a representative example, the graph is designed to include 15 vertices and 20 edges. To enable controlled comparison, a single edge is added to the original graph to form a new graph denoted as G_2 , with the original graph defined as G_1 .

Subsequently, the two graphs are visualized. To maintain consistency in vertex layout and facilitate direct comparison, both graphs are displayed within the same coordinate system. The newly added edge is explicitly highlighted to support subsequent comparison with the results obtained from the random walk method.

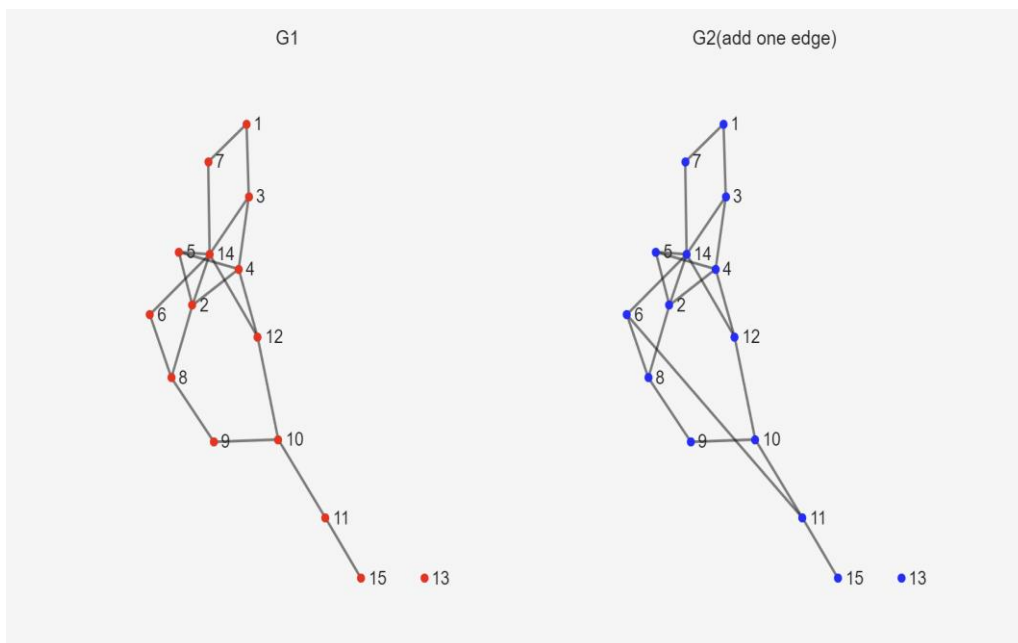


Figure 1. G_1 and G_2 (add one edge); $V=15$, $E=20$.

The newly added edge is clearly located between vertices 6 and 11.

Subsequently, PageRank and average return time are employed to characterize the graphs and detect structural differences. A total difference metric is used to identify vertices with significant changes.

Accuracy is adopted as the evaluation metric to verify whether the proposed method can accurately identify the changed vertices and the added edge. In addition, the vertices with the highest probability of edge relationship changes are visualized.

After executing the MATLAB program, the experimental results are as follows:

True different vertices: [6; 11]

Top-3 predicted vertices: [6; 11; 10]

Vertex identification accuracy: $2 / 2 = 100.00\%$

Edge localization accuracy: $1 / 1 = 100.00\%$

However, the proposed method still needs to be compared with the spectral matching method, which is viewed as authentic method.

We further compared the random walk-based method with the spectral matching method in terms of graph matching accuracy. For network graphs with the same set of vertices, the random walk method achieved an accuracy of 100%, significantly outperforming the spectral matching method, which yielded an accuracy of only 46.67%.

The matching result of Spectral Matching is as follow:

10	1
2	2
12	3
4	4
5	5
8	6
7	7
6	8
11	9
1	10
9	11
3	12
13	13
14	14
15	15

Figure 2. Matching result of spectral matching.

While the matching result of Random Walk is as follows:

1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15

Figure 3. Matching result of random walk.

In addition to accuracy, we also evaluated the computational efficiency of both methods by measuring their execution time. The random walk method completed the task in just 0.081504 seconds, whereas the spectral matching method required 0.175965 seconds.

To validate the robustness of our findings, we repeated the experiments on a larger graph with 25 vertices and 35 edges. Consistent with the previous results, the random walk method maintained its superior performance: it continued to achieve 100% accuracy while still exhibiting lower computational overhead than the spectral matching method, only 44.00%.

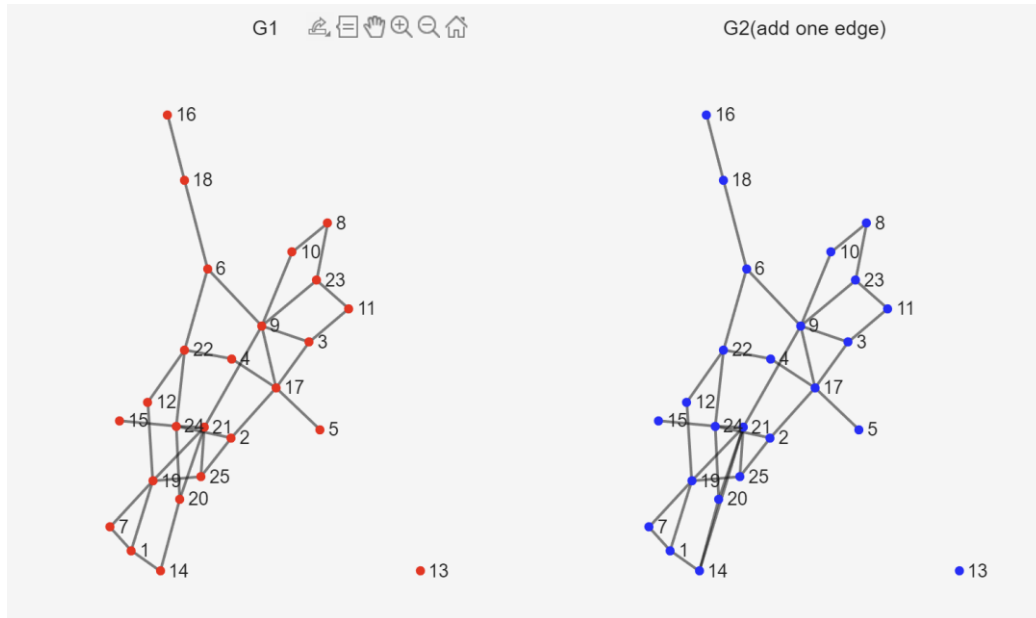


Figure 4. G1 and G2(add one edge); $V=25$, $E=35$.

3.2. Data Analysis

1. Vertex and Edge Localization Accuracy

Experimental results demonstrate that both vertex localization accuracy and edge localization accuracy reach 100%. This finding strongly validates the effectiveness of random walk in identifying structural discrepancies between graphs, and further reveals the considerable potential of random walk-based methods for graph comparison and difference detection tasks.

2. Graph Matching Accuracy

In our setting, the two graphs are highly similar (differing in only two edges) and share identical node labels. Since SM does not incorporate any node-level feature and relies exclusively on pairwise edge compatibility, the affinity matrix contains many equally strong signals for both correct and incorrect correspondences. Consequently, the principal eigenvector cannot reliably distinguish true matches, leading to a matching accuracy close to random.

In contrast, the proposed random walk method exploits node-level random walk characteristics (PageRank and mean return time), which are sensitive to local structural changes. By comparing these features between the two graphs, our method directly identifies the nodes affected by the edge alterations, achieving 100% node hit rate. This result demonstrates that for the task of change detection, local random walk features are more discriminative than global spectral matching.

3. Times

As the runtime of spectral matching is more than twice the runtime of the random walk approach, this result demonstrates that the random walk method holds a clear advantage in both accuracy and efficiency when compared to the established spectral matching technique.

4. Conclusion

Experimental results indicate that the random walk method achieves 100% vertex and edge localization accuracy, enabling effective identification of structural discrepancies between graphs. For graph matching tasks, Spectral Matching produces near-random performance owing to its

exclusive dependence on pairwise edge compatibility. In contrast, the proposed method leverages sensitive node-level random walk features to achieve a 100% node hit rate. In addition, the random walk approach exhibits a clear computational advantage, with a runtime less than half that of Spectral Matching. In summary, the random walk-based method surpasses Spectral Matching in both accuracy and efficiency for graph comparison and structural change detection tasks.

4. Future Applications: From Random Walk sensitivity to Big-Data Graph Models

The experiments presented in this work demonstrate that random walk features—specifically PageRank and mean return time—are highly sensitive to local structural changes between two graphs, achieving a 100% node hit rate in detecting the altered edges. This result confirms that RW captures fine-grained, node-level dynamics that directly reflect perturbations in graph structure.

However, the current method has several limitations. It relies on hand-crafted features, compares graphs only in a pairwise manner, and does not learn from data. When faced with large collections of graphs like thousands of evolving social networks, such an approach becomes inefficient and lacks adaptability.

Recent advances in big-data models, particularly graph neural networks, offer a promising direction to overcome these limitations. Graph neural networks can automatically learn structure-sensitive representations from data, process massive graph collections in parallel, and be trained end-to-end for tasks such as graph similarity learning, classification, or anomaly detection.

A natural extension of this work is to integrate random walk mechanisms into graph neural networks architectures. For example, random walk processes can be embedded as learnable layers that mimic multi-step propagation, allowing the model to automatically acquire sensitivity similar to that of PageRank or mean return time. Alternatively, the random walk-based difference scores computed in this study could serve as training labels for a graph neural network to predict structural dissimilarities across large graph datasets.

Such integration would enable the development of scalable and adaptive models capable of recognizing complex relationships among large numbers of graphs—ranging from similarity search and clustering to evolutionary pattern detection and anomaly identification—while preserving the structural sensitivity that random walks provide.

In summary, the sensitivity of random walks to structural changes, validated in this work, opens the door to more powerful, data-driven graph analysis frameworks that combine traditional random walk principles with modern big-data models.

5. Conclusion

In this study, the applications of random walk are first introduced. Subsequently, an experiment was conducted to verify that random walk can be employed to detect differences between two graphs. The experimental results demonstrate that random walk outperforms Spectral Matching in the case of two distinct graphs with the same number of vertices.

On this basis, it is concluded that when combined with big-data graph models, random walk performs more effectively in analyzing graphs with common node labels or structural discrepancies than in pure graph variation detection tasks. Furthermore, based on the experimental findings, it is suggested that random walk can be integrated with big-data graph models to tackle more complex tasks or process multiple graphs simultaneously.

Reference list

- [1] J. Yang, 2024, "Analysis of the Application for Random Walk in Computer Science, Physics and Mathematical Equations." International Division of No.2 High School of East China Normal University, Shanghai, China.

- [2] P. Haslum, 1999"Model Checking by Random Walk." Department of Computer Science, Linköping University.
- [3] U. Kang, M. Hebert, and S. Park, "Fast and scalable approximate spectral graph matching for correspondence problems," *Information Sciences*, vol. 220, pp. 306–318, Jan. 2013, doi: 10.1016/j.ins.2012.07.008.
- [4] Jin, D., Wang, R., Ge, M., He, D., Li, X., Lin, W., & Zhang, W. (2022). RAW-GNN: RANdom Walk Aggregation based Graph Neural Network. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI)*.
- [5] Dhillon, M., & Kataria, K. K. (2025). On elephant random walk with random memory. *arXiv*. <https://arxiv.org/abs/2501.12866>
- [6] Pearson K. The problem of the Random Walk. *Nature News*, Nature Publishing Group, 1905, 7: 1.
- [7] Ng, P. S., & Suo, J. (2022). How can random walk be applied to analysis stock. *Academic Journal of Mathematical Sciences*, 3(2). <https://doi.org/10.25236/ajms.2022.030205>
- [8] Cox, John C., Stephen A. Ross, and Mark Rubinstein. "Option pricing: A simplified approach." *Journal of financial Economics* 7.3 (1978): 229-263.
- [9] Bera, S. K., & Seshadhri, C. (2020). How to count triangles, without seeing the whole graph. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 431–441). Association for Computing Machinery. <https://doi.org/10.1145/3394486.3403073>
- [10] Leordeanu, M., & Hebert, M. (2005). A spectral technique for correspondence problems using pairwise constraints. *ICCV*.