Approximate Triangle Count and Clustering Coefficient

Siddharth Bhatia
BITS, Pilani

ABSTRACT

Two important metrics used to characterise a graph are its triangle count and clustering coefficient. In this paper, we present methods to approximate these metrics for graphs.

ACM Reference Format:
Siddharth Bhatia. 2018. Approximate Triangle Count and Clustering Coefficient. In *SIGMOD'18: 2018 International Conference on Management of Data, June 10–15, 2018, Houston, TX, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3183713.3183715

1 INTRODUCTION

The need for analytic tools to describe and understand graphs is stronger than ever. The triangle count (TC) and the clustering coefficient (CC) are the two most commonly used metrics to characterise a graph. They both give an intuition of the community structure. In this paper, we extend the methods presented in [3] to allow the approximation of the TC and CC of graphs. These methods are based on wedge sampling. A wedge is a path of length 2: a pair of edges sharing a vertex. Here, we make the assumption that the graph is stored in CSR format which is one of the most commonly used format for sparse graphs. We describe the method in Section 2 and its performance in Section 3.

2 METHOD

2.1 Triangle Count and Clustering Coefficient

A triangle in an undirected graph is a set of three vertices such that any two of them are connected. A wedge is a set of three vertices such that one of them, the center, is connected to the other two. The triangle count is the number of triangles and the global clustering coeffient is the ratio of triangles and wedges. Graph is stored in CSR format because it allows us not to process the entire graph before sampling. Also the possible atomic operations are node sampling, edge sampling, node sampling with distribution proportional to degree, and finding the degree of a vertex in $O(1)$.

Symbols: V : number of vertices, E: number of edges, W: number of wedges, d_v: degree of vertex v, S_0: number of vertices without an edge ($d_v = 0$), S_1: number of vertices with a single edge ($d_v = 1$), $W_v = \binom{d_v}{2}$: set of wedges centred on vertex v, $v(w)$: the vertex in the center of wedge w, $c(w)$: whether wedge is closed. 1 if it is (there is a triangle) and 0 otherwise, $TC = \sum_{w \in W} c(w)/3$: triangle count, $CC = \frac{TC}{W}$: clustering coefficient

SIGMOD'18, June 10–15, 2018, Houston, TX, USA

ACM ISBN 978-1-4503-4703-7/18/06.
https://doi.org/10.1145/3183713.3183715

2.2 Approximating Triangle Count and Clustering Coefficients

Three methods have been considered to estimate the triangle counts. Each of them uses a different distribution to sample vertices for which we randomly generate a wedge and use importance sampling if necessary. Importance sampling works by reweighing the samples. Simply put, if you're looking for the average value of $f(x)$ under distribution $p(x) : E_p(f(x))$, and we can sample from distribution $q(x)$, we note that $E_q(\frac{p(x)f(x)}{q(x)}) = E_p(f(x))$. In other words, we find the average value of $\frac{p(x)f(x)}{q(x)}$ under distribution $q(x)$. Because calculating the clustering coefficient from the triangle count is so simple, we are only considering the triangle count at the moment.

2.2.1 Uniform Wedge. The first method is based on the idea of sampling the wedges uniformly and then projecting this result to the whole of the graph. It is easy to calculate how many wedges a vertex has since it is the number of ways we can choose two of its neighbors. Because we are using the CSR format, we already know the accumulative indicies. From that we can calculate the accumulative wedgecounts.

Algorithm 1 Uniform Wedge

```
1: TotalWedges ← 0
2: for v ∈ {0, ..., V − 1} do
3:     AccWedgeCount[v] = TotalWedges
4:     TotalWedges += d_v*(d_v−1)/2
5: end for
6: sum = 0
7: for i ∈ {0, ..., SampleSize − 1} do
8:     r = RandomNumber(0, TotalWedges)
9:     index = binarySearch(r, AccWedgeCount)
10:    w = GenerateRandomWedge(index)
11:    sum += c(w)
12: end for
13: return TotalWedges*sum / 3*SampleSize
```

If we pick a number uniformly from the interval $[0, W)$ and find the corresponding vertex using a binary search on the accumulative wedges then the probability of choosing a given vertex v is $\frac{W_v}{W}$ and therefore the probability of choosing a given wedge w with center v is $p(w) = \frac{1}{W}$. Because the distribution is uniform, we do not need to use importance sampling. The triangle count estimate is $TC = \frac{WE_p(c(w))}{3}$. The division by 3 is necessary because we count each triangle three times as each vertex is a possible center of a wedge.

2.2.2 Uniform Edge. In the second method we sample the edges uniformly. Choosing a number randomly in the interval $[0, E)$ corresponds to an edge from the edge list. Looking this up in the accumulative indices we can find v. We have to repick if v has only a single edge because then there is no wedge to sample. We are choosing a vertex v with probabilty $\frac{d_v}{V-S_1}$. Then choosing w such that v is the center of w corresponds to $q(w) = \frac{d_{v(w)}}{W_{v(w)}(V-S_1)} = \frac{2}{(d_{v(w)}-1)(V-S_1)}$.

Algorithm 2 Uniform Edge

$S_1Estimate \leftarrow 0$
for $i \in \{0, \ldots, SampleSize - 1\}$ **do**
  $r = RandomNumber(0, V)$
  if $d_r = 1$ **then**
    $S_1Estimate$++
  end if
end for
$S_1Estimate = \frac{V * S_1Estimate}{SampleSize}$
$sum = 0$
for $i \in \{0, \ldots, SampleSize - 1\}$ **do**
  repeat
    $r = RandomNumber(0, E)$
    $index = search(r, AccIndices)$
  until $d_{index} > 1$
  $w = GenerateRandomWedge(index)$
  sum += $c(w)(d_{index} - 1)$
end for
$return \frac{(2E - S_1Estimate)(sum)}{6 * SampleSize}$

Because the distribution is not uniform on the wedges we need to apply a weight $\frac{d_{v(w)}-1}{2}$ to the samples according to the method of importance sampling. The triangle count estimate is $TC = \frac{2E-S_1}{6} E_r((d_{v(w)} - 1)c(w))$. Of course for that we need S_1 so we need to estimate that too. The benefit of this method is that it requires no precomputation of the accumulative wedgecounts but we lose some accuracy with the non-uniform sampling.

2.2.3 Uniform Vertex. In the third method we sample the vertices uniformly. Choosing a number randomly in the interval $[0, V)$ corresponds to a vertex v. We have to repick if v has no or only a single edge because then there is no wedge to sample. We are choosing a vertex v with probabilty $\frac{1}{V-S_1-S_0}$. Regarding the wedges, that corresponds to $r(w) = \frac{1}{W_{v(w)}(V-S_1-S_0)}$.

Algorithm 3 Uniform Vertex

$SEstimate \leftarrow 0$
for $i \in \{0, \ldots, SampleSize - 1\}$ **do**
  $r = RandomNumber(0, V)$
  if $d_r = 1$ or $d_r = 0$ **then**
    $SEstimate$++
  end if
end for
$SEstimate = \frac{V * SEstimate}{SampleSize}$
$sum = 0$
for $i \in \{0, \ldots, SampleSize - 1\}$ **do**
  repeat
    $index \leftarrow RandomNumber(0, V)$
  until $d_{index} > 1$
  $w = GenerateRandomWedge(index)$
  sum += $c(w) \frac{(d_{index}-1)(d_{index}-1)}{2}$
end for
$return \frac{(V - SEstimate)(sum)}{3 * SampleSize}$

Because the distribution is not uniform on the wedges we need to apply a weight $(W_{v(w)})$ to the samples according to the method of importance sampling. The triangle count estimate is $TC = \frac{V-S_1-S_0}{3} E_r(W_{v(w)}c(w))$. This method also has the benefit of no precomputation and requires no access to the edge list for the sampling. In practice the disadvantages outweight the advantadges because the difference between the triangle counts of higher degree verticies and lower degree vericies leads to intolerable inaccuracy therefore this method was not extensively tested.

Table 1: Graphs used for the experiments

Graphs	Nodes	Edges	Triangles	Size	AvDeg	STD	MaxDeg	Source
Twitter	61.6M	1.5B	34,824,916,864	9.4GB	57.7	402	2,997,487	[2]
RMAT-24	16.7M	268.4M	10,489,616,353	2.1GB	56.5	555	175,007	[1]
RMAT-25	33.6M	536.9M	23,284,863,734	4.2GB	58.8	527	273,738	[1]
RMAT-26	67.1M	1.1B	51,559,452,522	8.4GB	61.2	632	430,269	[1]
RMAT-27	134.2M	2.1B	114,007,006,286	17GB	63.6	601	676,199	[1]

3 EVALUATION

For evaluation purposes, we conducted experiments on a 24- core machine. The graphs used are summarized in Table 1. Our experiments showed Uniform Vertex to be inferior to the other methods in almost all the cases. This can be explained by the extremely high maximum degrees of our datasets (shown in Table 1.) Using the Uniform Vertex method, the result is very sensitive on how many times the high degree vertices are picked and, since these are very few, a lot of samples are required for high accuracy. The Uniform Edge performs better than the Uniform Vertex as it is closer to the original distribution of the Unifrom Wedge and, also, handles the problem of high degrees better. This is because the high degree nodes are lucky to be picked since they have many edges. Note, however, that it is not as good as the Uniform Wedge at this because the number of wedges go as squared the degree and, thus, those vertices are picked almost exclusively. Another shortcoming of both of these methods is that it is very difficult to provide confidence bounds, as the distribution needs scaling. We evaluated the convergence of Uniform Wedge method with increasing sample sizes and found that the accuracy of the method does not depend on the size of the graph. Instead, it depends on the global clustering coefficient; the closer it is to 0.5, the quicker the convergence. In order to be able to reason about graphs with unknown number of triangles, we provide confidence bounds as well as the estimated number. Figure 1 shows the bounds for Twitter at 10^5 and 10^6 samples. There is only one case where the triangle number is outside the 90% interval, and thus the bounds are accurate.

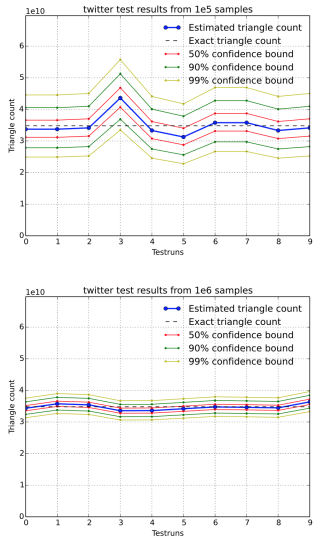

Figure 1: Confidence bounds on Twitter dataset

REFERENCES

[1] Deepayan Chakrabarti, Yiping Zhan, and Christos Faloutsos. R-mat: A recursive model for graph mining. In *Proceedings of the 2004 SIAM International Conference on Data Mining*, pages 442–446. SIAM, 2004.

[2] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM, 2010.

[3] Comandur Seshadhri, Ali Pinar, and Tamara G Kolda. Triadic measures on graphs: The power of wedge sampling. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 10–18. SIAM, 2013.

[4] Bin Wu, Ke Yi, and Zhenguo Li. Counting triangles in large graphs by random sampling. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2013–2026, 2016.