Analysis of the Recommender System of an Online Retailer

Alexander Cam Liu
Anupriya Srivastava

Introduction

The long tailed phenomenon, especially as it relates to the e-commerce industry is a topic of considerable interest and research. The traditional brick and mortar stores have to rely upon a fixed retail space, which restricts their ability to market specialized niche products. But their online counterparts have no such restriction and are therefore able to carry specialized merchandize that caters to niche clientele [1]. As some online retailers look to take further opportunity of this competitive advantage, we see a trend where more and more of their revenue is coming from low volume merchandize. This has led to considerable debate about the exact extent to which this phenomenon is subverting traditional retail models.

While there is already quite a bit of research that looks at the hard financial numbers, we will attempt to gain insight be looking at the recommender system graph of one particular online retailer. While the exact details of the recommender system algorithm are proprietary, we nevertheless start with the assumption that the recommender system is self evolving model that finds an optimum steady state yielding maximum recommendations to hits conversion. With this assumption, we can reach some prima facie conclusions. If the market is heavily segmented into niche customer groups, then we would expect to see a lot of small clusters, relating to each specialized segment, which in turn means that the size of the hubs will remain relatively contained. If on the other hand, the market conforms to the traditional models where a few high volume products make the bulk of the sales, then we would expect certain key products to form the predominant hubs with large maximum hub sizes. This is the rich gets richer phenomenon that we see in all different fields of study.

In order to do a more in depth analysis on this network, we will study the network in a systematic manner using the networking tools at our disposal. We start by clustering the network to observe the overarching trends that affect the nodes distribution, and make initial guesses about the underlying structure of the graph. We use this initial guess to model the degree distribution of the graph, and find the best fit curves for it. Finally we try to determine the root causes for deviations observed from predictive models, by analyzing the dynamic growth of the graph over time. In the end, we try to determine if our analysis provides any insights into the structure of the consumer base of this online retailer.

Datasets and Methodology

In this project, we use two separate Recommender System data sets from Amazon Inc, that correspond to the same general categories of items but different time frames. Network was

collected by crawling Amazon website. It is based on "*Customers Who Bought This Item Also Bought"* feature of the Amazon website. If a product *i* is frequently co-purchased with product *j*, the graph contains a directed edge from *i* to *j*. The network was obtained from Stanford's SNAP portal.

Dataset 1: **Amazon product co-purchasing network from March 2 2003.**

- https://snap.stanford.edu/data/amazon0302.html
- Nodes: 262111
- Edges: 1234877

Dataset 2: **Amazon product co-purchasing network from March 12 2003.**

- https://snap.stanford.edu/data/amazon0312.html
- Nodes: 400727
- Edges: 3200440

The primary tools used are Gephi and R Studio with iGraph and poweRlaw packages. Gephi provided the visual representation capabilities, while R Studio provided the analytical tools for more in depth analysis.

Step 1: Clustering

Dataset 1:

We determined that the modularity coefficient of Dataset 1 is **0.893**. This is a relatively high modularity which implies the presence of tightly connected *clusters* in the network.

Fig.1 Modularity of big *hubs* in the Dataset 1 Network

Fig. 1 shows the *clusters* with the highest density in the Dataset 1 Network. We note that there are only 4 main clusters, which we can assume belong to 4 different types of products (or categories in Amazon). A large *cluster* can mean that those items are typically recommended together, and would probably be bought together by the customers. A user would typically buy products together in order to achieve something. For example, a user might want to build a computer and will to buy several items from the computer parts category in order to build it.

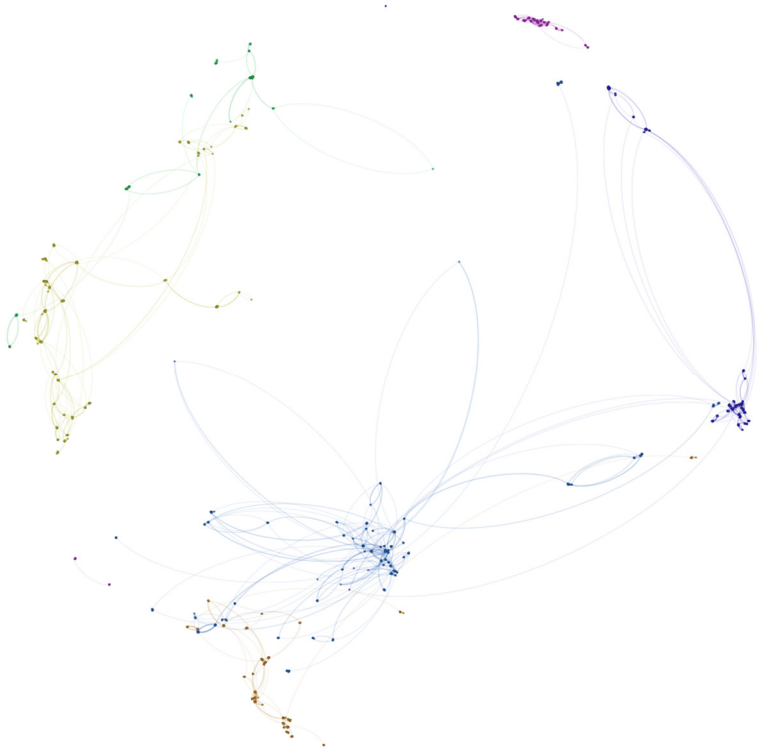

Fig.2 Modularity of small *clusters* in the Dataset 1 Network

Fig. 2 shows the *clusters* with smaller density. These refer to sub-categories of products that are bought together more frequently. These could symbolize specialized products that cater to niche clients.

Dataset 2:

The modularity of Dataset 2 is **0.867** which is almost the same as the previous dataset. Fig. 3 shows the two biggest *clusters* from the data set. Only the top two clusters are shown since the network has over 3 million edges.

Fig.3 Modularity of two largest *clusters* in the Dataset 2 Network

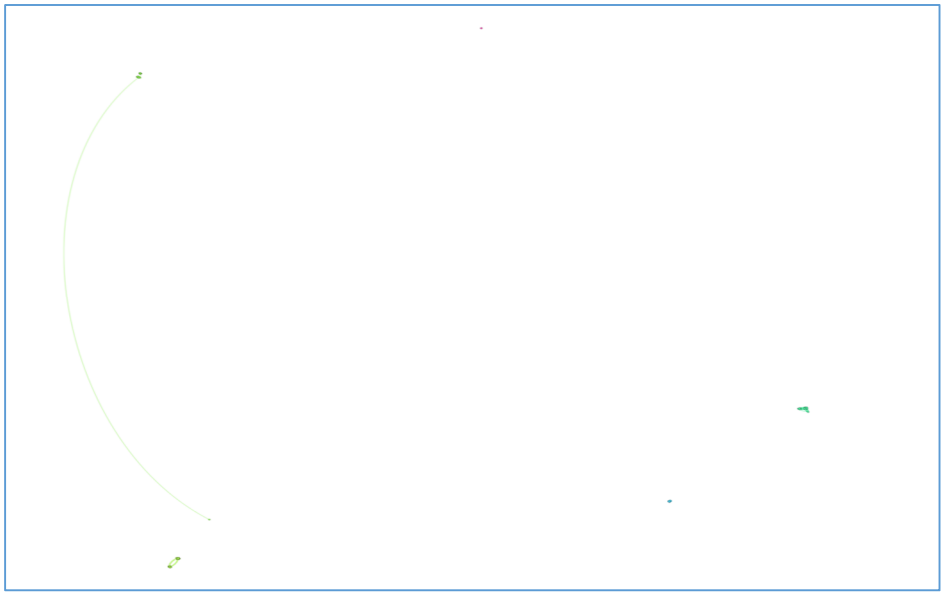

Fig.4 Modularity of small *hubs* in the Dataset 2 Network

The main observation that pops put from the clustering diagram is that there are a small number of high density clusters, containing several medium to high degree hubs within them. We don't see a large number of disjointed small clusters of niche products from the preliminary visual analysis. This observation in itself does not mean that such niches don't exist within the larger clusters; it only means that such niches are not isolated in themselves.

Step 2:

In order to determine the exact nature of the graph, we look at the degree distribution of the network. This allows us to determine the best fit model for the network, and in turn obtain insights based on observations from similar networks.

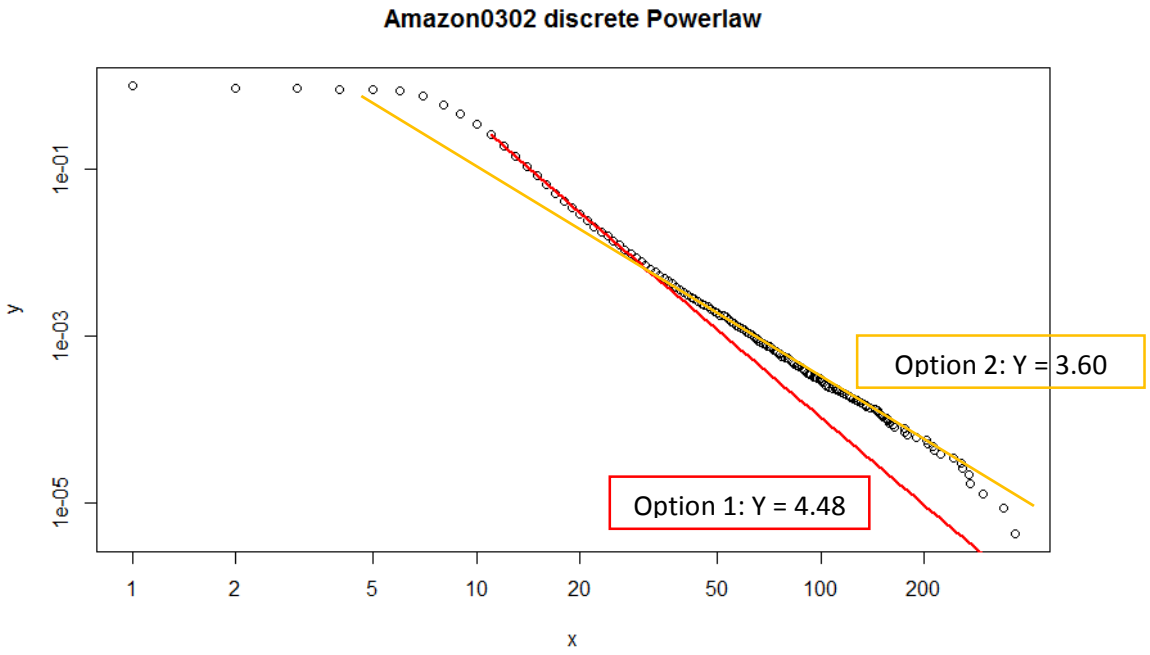

Fig.5 Log log plot of the Dataset 1 Network

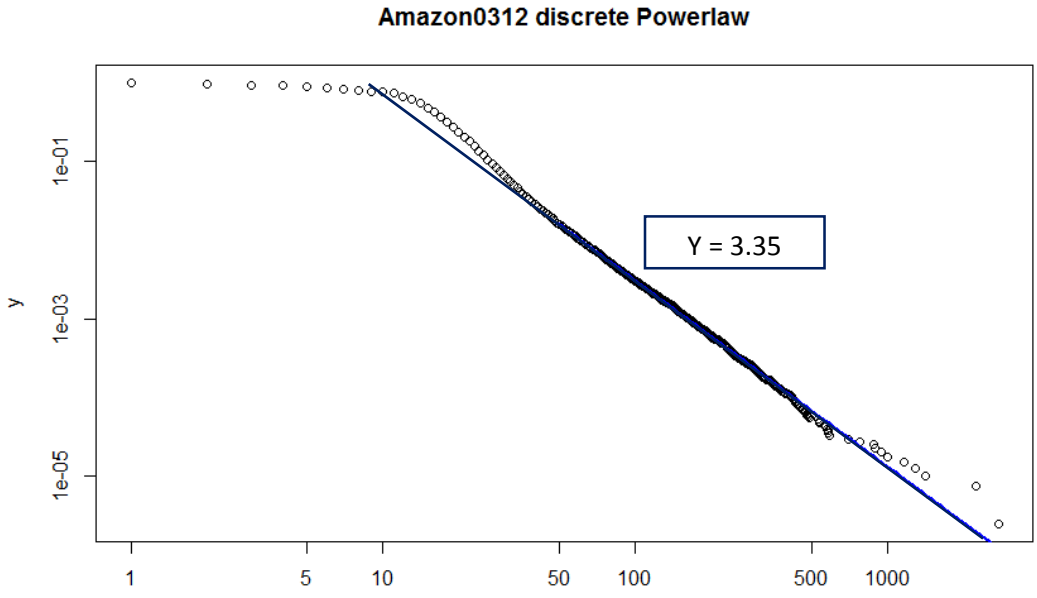

Fig.6 Log log plot of the Dataset 2 Network

Fig 5 and 6 show the log-log plot of the degree distribution for Dataset 1 and Dataset 2 respectively. We note that both plots conform to the power-law form with a low k saturation. This leads us to speculate that it belongs to the class of scale free networks, with a few deviations. The first dataset also shows two possibilities for the exponent value of the power law line. If we assume option 2, the initial attraction theory is a great candidate for explaining most of the deviations [9, Ch 6]. The low k saturation can be explained by the theory of initial attraction, which predicts that all nodes end up gaining initial popularity due to the novelty factor. The small bump right before the start of the saturation region is also expected from the initial attraction theory since the extremely low k nodes get pushed up to the medium k region [9, Ch 5]. Finally we also note that the power law exponent > 3 can is also explained by this initial attraction theory.

However, it is also intriguing to consider option 1, and look at what might be causing the high k nodes to flare/extend away from the power-law curve. To study this further, we look at the other parameters associated with these datasets.

Graph	Nodes	Exponent	<k>	$<k^2>$	k_max	Distance	Diameter (90%)
Amazon 0302	262111	4.48	4.50	114.7	361	Diameter = 32 <d> ~ 10	11
Scale Free (theoretical)	262111	4.48	4.50	N/A	$N^{1/(Y-1)}$ ~ 36	<d> = ln N ~ 12.47	
Amazon 0312	400727	3.35	7.98	505.9	2747	Diameter = 18 <d> ~ 6	7.6
Scale Free (theoretical)	400727	3.35	7.98	N/A	$N^{1/(Y-1)}$ ~ 242	<d> = ln N ~ 12.90	

Fig.7 Key parameters of both the datasets

In the table above we show the calculated parameters for the two datasets (using option 1 for Dataset 1), along with the theoretical parameters of a Power Law/ Scale Free graph of same size. The first deviation that sticks out is the extremely high max_k value, which is more than an order of magnitude greater than expected theoretical values. This is actually consistent with the flare/extend out from the ideal power law line that we see in Option 1 of Dataset 2, and also in Dataset 2 (but to a lesser degree). The other huge anomaly that seems to defy explanation in the context of scale free networks is that the Diameter of Dataset 2 decreases by a factor of two, while the size of the Dataset 2 had increased by about the same factor. Finally we also note that <k>, average links value, also increased by close to a factor of two, which is unexpected.

One prima facie explanation can be that as the network experience accelerated growth, where its links grew faster than the size of the graph. This would explain the increase in the average links size <k>, and the exponent value greater than 3 [9, Ch 6]. But is still doesn't fully explain the drastic decrease in the diameter or extremely high max_k values.

The other explanation that we would like to explore is that as the network grew at an accelerated pace, it also developed a superhubs more in line with the winner takes all paradigm. This would explain both the extremely large k_max, and reduced average distance between nodes. This would also be consistent with the deviation in the degree distribution plot that pushed high k nodes farther away from the power law plot.

However, to truly substantiate this explanation, we will have to explore how the network grows dynamically in order to determine if it shows any tendency towards super linear preferential attachment [9, Ch 5].

Step 3

In this step we plot the delta_k between the two plots in order to determine the growth rate of the node degrees. We first note that our raw data that does not include the name/ product ID of the nodes; therefore there is no way to directly measure a change in degree of each node. Instead we heuristically determine this indirectly from the cumulative degree distribution plots. (Annex A shows the exact methodology used to plot this information, including the R script).

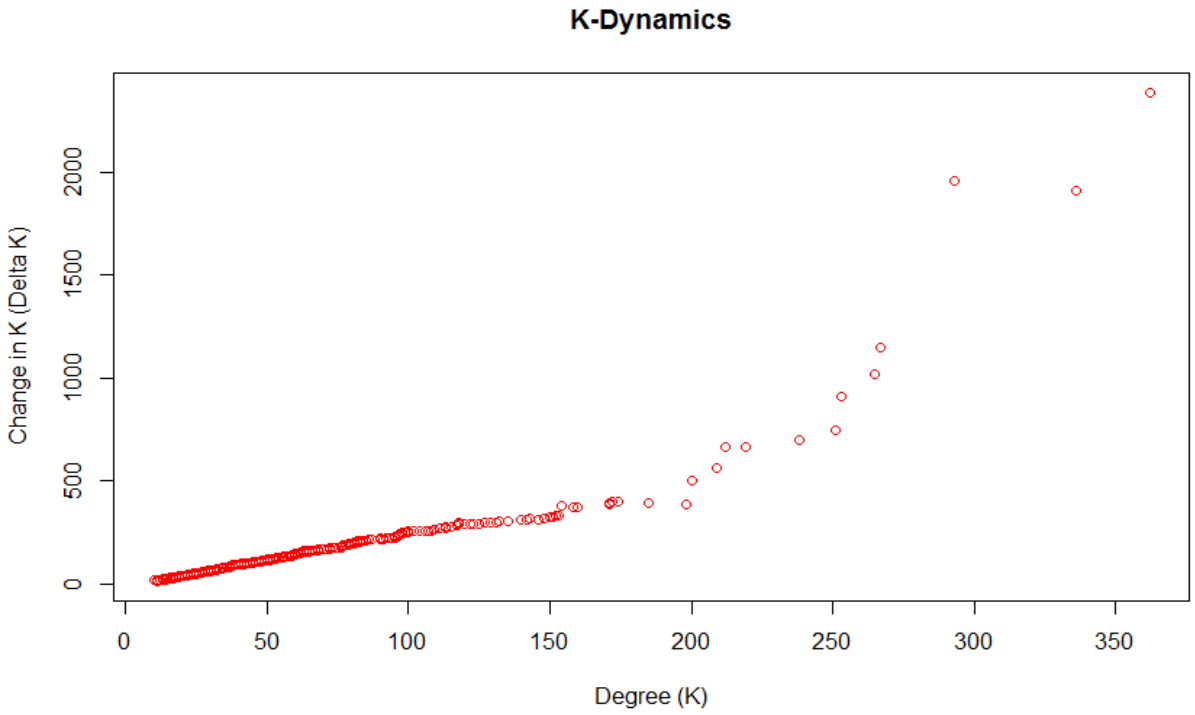

Fig.8 Change in degree over time

At first glance we see that for the most part change in K is linear in K. This should indicate that there is linear preferential attachment at play, which results in scale free/ power law category of networks [9, Ch 5]. However, when we look at the high k region, we notice that the plot takes up a roughly $x^{1.5}$ polynomial form.

This leads us to conclude that while linear preferential attachment is at play for majority of the nodes, superlinear preferential attachment in the high_k region is creating a few super hubs. In

this case, we note that the top three nodes have a node count which is one order of magnitude more than the average node count, and three times more than the next highest node count. This could possibly qualify them as superhubs. In relation to the recommender system, these nodes could be the nodes that overlap certain categories, or simply be high grossing items that the vendor wants to bring to the customers attention.

With this understanding of the dynamics of the growth of the network, we believe that the best explanation with regard to all the observed characteristics is an accelerated growth pattern with superlinear preference for certain key nodes that ended up serving the role of supernodes in this network. These observations speak mostly in favor of the network supporting rich gets richer phenomenon, where certain popular/ high grossing products end up getting the disproportionately high number of recommendations.

However we still can't make a definitive statement about the presence of mini niche clusters due to their integrated nature. We can only make comparative statement by looking at the proportion of links that are divided between the high k and low k regions. Looking at the larger Dataset 2, we note that top 20% of the highest k nodes received about 50% of the total recommendations. While this definitely a disproportionately high amount, it is still not as egregious as the 20/80 split noted in more traditional markets [1]. The difference certainly could be due to the niche products building highly dense mini networks and siphoning away a share of the recommendations from larger nodes. We also note that the bump region (right before the beginning of the saturation zone) also has a disproportionately large number of recommendations, with 20% of the nodes claiming 50% of the remaining 50% of the recommendations (25% of the total recommendations). Therefore this bump could also indicate the presence of dense niche clusters.

Conclusion

We set out to determine if we can use the recommender system network to gain any insight into the long tail characteristic of several online vendors. In order to analyze our network, performed a cluster analysis, degree distribution plot analysis and preferential growth analysis. In the end we noted that while there was strong support for the presence of niche clusters, they did not prevent the graph from retaining its scale free nature and perpetuating the rich gets richer phenomenon. In fact we saw the presence of very large degree hubs (potentially superhubs), which were the result of superlinear preference for certain key nodes in the network.

One simple explanation for the presence of both of these phenomenon at the same time can be the artificial nature of the recommender systems which are prone to complex manipulations by the vendors to serve as advertisement vehicles as well. As a result, the vendor would include recommendations for their high grossing/ profitable products in addition to the niche product recommendations relevant to the perspective customer.

As a result we see a recommender system that maintains a level of duality in which it can support niche products while at the same time retain the rich-gets-richer profile and even allow for the presence of extremely large k nodes (potentially superhubs) that we have come to expect from traditional marketing models.

References:

1. The dynamics of viral marketing. –By J Leskovec .ACM transactions on the web. Vol 1 no. 1 article 5. 2007

2. Blockbuster culture's next rise or fall: The effect of recommender systems on sales diversity .-By Daniel Fleder Kartik Hosanagar The Wharton School, University of Pennsylvania, Philadelphia, PA

3. Empirical analysis of predictive algorithms for collaborative filter. –By Breese, J., D.,Heckerman, and C. Kadie. 1998. ing. 14th Conference on Uncertainty in Artificial Intelligence.

4. Recommender systems in e-commerce. –By Schafer, J., J. Konstan, and J. Riedl. 1999. In Proceedings of the ACM Conference on Electronic Commerce, p. 158-166.

5. A Survey on Approaches for More Accurate and Diverse Recommendations. –By Sangeetha G M and Mr. Prasanna Kumar M. International Journal of Advanced Research in Computer Science and Software Engineering. Volume 2, Issue 11, November 2012

6. Recommenderlab: A Framework for Developing and

7. Testing Recommendation Algorithms- by Michael Hahsler. Southern Methodist University

8. Easley & Kleinberg Networks, Crowds and Markets.

9. Barabasi Network Science Book

Appendix A- K Dynamics

The Delta k plot was obtained using the assumption that most new nodes would have a relatively low k count, and therefore if we order the list of nodes by their degree, we can (on average) assume a one to one correspondence between nodes of the two networks at the high k end of the graph. This is to say that the highest k value node at time t1 remains the highest k value node at time t2, similarly the 2nd highest k value node at time t1 remains the 2nd highest k value node at time t2, and so on till we start reaching the low k regions.

The script for obtaining the change in degree (Delta K) [Fig 8]

```
g1 <- read.graph("C:/Users/Anupriya/Desktop/spring/622-
ds/project/Amazon0302m.net", format="pajek")

g2 <- read.graph("C:/Users/Anupriya/Desktop/spring/622-
ds/project/Amazon0312.net", format="pajek")

k_min <- 5

m<-determine_delta_high_end(g1,g2,k_min, 20000, 1, .5)


plot(m[,1], m[,2],col="red", main="K-Dynamics",

     xlab="Degree (K)", ylab="Change in K (Delta K)")


deg_g1 <- degree(g1)

deg_g2 <- degree(g1)


g1_dd <- degree.distribution(g1, cumulative = F, mode = "in")

g2_dd <- degree.distribution(g2, cumulative = F, mode = "in")


determine_delta_high_end <- function(g1, g2, k_min,
index_jump_counter_global, index_jump_counter_outer, index_jump_amount){

  # g2 is the more dense graph with higher links per node

  #k_min is the kvalue where graph begins to straighten out

  # index values help determine which k value this algorithm covers

  #diff_matrix: matrix where data will be stored; this value is returned
```

```
  diff_matrix <- matrix(nrow = 0, ncol = 3, dimnames = list(c(), c("k_value",
"k_delta", "cum_k_delta")))


  g1_ddc <- degree.distribution(g1, cumulative = TRUE, mode = "in")

  g2_ddc <- degree.distribution(g2, cumulative = TRUE, mode = "in")

  g1_dd <- degree.distribution(g1, cumulative = F, mode = "in")

  g2_dd <- degree.distribution(g2, cumulative = F, mode = "in")


  # We will start by looking at the nodes at the high K end

  g1_index = length(g1_ddc)

  g2_index = length(g2_ddc)

  g1_size = vcount(g1)

  g2_size = vcount(g2)


  g1_nodes_basket.count <- 0

  g2_nodes_basket.count <- 0

  g1_ddc_last_value <- 0

  g2_ddc_last_value <- 0

  #index_jump_counter_global <- 330

  index_jump_counter <- index_jump_counter_global

  #index_jump_counter_outer <- 4

  #index_jump_amount <- .5


  while (g1_index > k_min && g2_index > k_min && index_jump_counter_outer >
0){

    # Find the next highest k node


    if (g1_nodes_basket.count == 0){

      g1_ddc_new_value <-g1_ddc[g1_index]

      while (g1_ddc_last_value>=g1_ddc_new_value){

        g1_index<-g1_index  -1

        g1_ddc_new_value <-g1_ddc[g1_index]

      }
```

```
      g1_nodes_basket.count <- (g1_ddc_new_value- g1_ddc_last_value) *
g1_size

      g1_nodes_basket.count <-round(g1_nodes_basket.count)

      #print (c("g1 node basket count", g1_nodes_basket.count,
g1_ddc_new_value, g1_ddc_last_value))

      g1_ddc_last_value<-g1_ddc_new_value

    }

    if (g2_nodes_basket.count == 0){

      g2_ddc_new_value <-g2_ddc[g2_index]

      while (g2_ddc_last_value>=g2_ddc_new_value){

        g2_index<-g2_index - 1

        g2_ddc_new_value<- g2_ddc[g2_index]

      }

      g2_nodes_basket.count <- (g2_ddc_new_value- g2_ddc_last_value) *
g2_size

      #print (c("g2 node basket count", g2_nodes_basket.count,
g2_ddc_new_value, g2_ddc_last_value))

      g2_nodes_basket.count <-round(g2_nodes_basket.count)

      g2_ddc_last_value <- g2_ddc_new_value

    }

    diff_matrix <- rbind( diff_matrix, c(g1_index, g2_index-g1_index, 0))

    g1_nodes_basket.count <- g1_nodes_basket.count -1


    g2_nodes_basket.count <- g2_nodes_basket.count-1

    #print (c("g2 node basket count after minus", g2_nodes_basket.count))

    #print (c("g1 node basket count after minus", g1_nodes_basket.count))

    index_jump_counter <- index_jump_counter - 1

    if (index_jump_counter == 0){

      #print (" inside last if")

      g1_index <- round(g1_index * index_jump_amount)

      g1_ddc_new_value <-g1_ddc[g1_index]

      g1_ddc_last_value <-g1_ddc[g1_index+1]

      while (g1_ddc_last_value>=g1_ddc_new_value){

        g1_index<-g1_index  -1
```

```
        g1_ddc_new_value <-g1_ddc[g1_index]
      }

      g2_target<-round(g1_ddc[g1_index] * g1_size)
      while (round(g2_ddc[g2_index]*g2_size) < g2_target){
        g2_index <- g2_index - 1
        #print(g2_ddc[g2_index])
      }
      g2_nodes_basket.count <- 0
      g1_nodes_basket.count <- 0
      g1_ddc_last_value <- g1_ddc[g1_index+1]
      g2_ddc_last_value <- g2_ddc[g2_index+1]
      index_jump_counter <- index_jump_counter_global
      index_jump_counter_outer <- index_jump_counter_outer - 1
    }

  }
  return (diff_matrix)
}
```