Assignment 7 for Large Scale Data Mining

Name: Zijian Zhang
Matrikelnr.: 3184680

June 14, 2016

1

1.1

$$\begin{aligned}
p(heads = h, tails = t|p) &= \binom{h+t}{h} p^h (1-p)^t \\
L(p|heads = h, tails = t) &= p(heads = h, tails = t|p) \\
\ln(L) &= C + h\ln(p) + t\ln(1-p) \\
\frac{d(\ln(L))}{dp} &= 0 \\
\frac{h}{p} - \frac{t}{1-p} &= 0 \\
p &= \frac{h}{h+t}
\end{aligned}$$

1.2

$E(p) = 0.5 * 90\% + 0.1 * 10\% = 0.46$, so the value of h and t should satisfy $\frac{h}{h+t}$>0.46.

2

2.1

$$F_{vC} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$A_{(u,v)} = \begin{pmatrix} 0 & \varepsilon & \varepsilon & 1-e^{-1} \\ \varepsilon & 0 & 1-e^(-1) & \varepsilon \\ \varepsilon & 1-e^(-1) & 0 & \varepsilon \\ 1-e^(-1) & \varepsilon & \varepsilon & 0 \end{pmatrix}$$

$$L = \varepsilon^3(1-\varepsilon)e^{-1}(1-e^{-1})$$

2.2

$$F_{vC} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix}$$

$$A_{(u,v)} = \begin{pmatrix} 0 & 1-e^{-2} & 1-e^{-2} & 1-e^{-1} \\ 1-e^{-2} & 0 & 1-e^{-2} & 1-e^{-1} \\ 1-e^{-2} & 1-e^{-2} & 0 & 1-e^{-1} \\ 1-e^{-1} & 1-e^{-1} & 1-e^{-1} & 0 \end{pmatrix}$$

$$L = (1-e^{-2})^2(1-e^{-1})^2e^{-3}$$

3

Because the edges are generally picked independently with probability p, the probability that vertex u and v are connected is p, and that they are with probability $1-p$ not connected. Thus the likelihood of generating this graph is:

$$L(G|\Theta) = p^4(1-p)^2$$

The maximum likelihood occurs on probability:

$$\frac{d\ln(L(G|\Theta))}{p} = \frac{4}{p} - \frac{2}{1-p} = 0$$

$$p = \frac{2}{3}$$

Meanwile, the likelihood of this graph is:

$$L(G|\Theta) = (\frac{2}{3})^4(1-\frac{2}{3})^2 = \frac{16}{729}$$

4

4.1

Clustering coefficient for each node of a clique is:

$$cc(v) = \frac{\#\Delta v}{\binom{n-1}{2}} = 1$$

4.2

Clustering coefficient for each node of a complete bipartite graph is:
for left set (with vertex number of l):

$$cc(v) = \frac{\#\Delta v}{\binom{l+r-1}{2}} = \frac{\binom{r}{2}}{\binom{l+r-1}{2}} = \frac{r(r-1)}{(r+l-1)(r+l-2)}$$

for right set (with vertex number of r):

$$cc(v) = \frac{\#\Delta v}{\binom{l+r-1}{2}} = \frac{\binom{l}{2}}{\binom{l+r-1}{2}} = \frac{l(l-1)}{(l+r-1)(l+r-2)}$$

4.3

p neighbors of node A has the probability being connected with an edge with p, so the expected triangle formed using node A is $\binom{p}{2}p$. Let the graph G contain n vertex, the expected coefficient of node A should be $\frac{\binom{p}{2}p}{\binom{n}{2}}$.

5

5.1

$\sqrt{m} = \sqrt{17} \equiv 4.12$, so the minimum degree for a node to be considered a heavy hitter should be $\lfloor 4.12 \rfloor = 5$

5.2

Heavy hitters are $\{U_1, U_2, T_2, W_2\}$

5.3

Triangles $\{\Delta U_1U_2T_2, \Delta U_1U_2W_2, \Delta U_1T_2W_2, \Delta U_2T_2W_2\}$ are heavy-hitter triangles.

6

6.1

Map input: $<, \Gamma(u)>$, output:$<, (v_1, v_2)>$ where $v_1, v_2 \in \Gamma(u)$
Reduce input: $<(v_1, v_2), U>$, in each machine the (v_1, v_2) pairs are the same and $\forall u \in U : E(u, v_1) \wedge E(u, v_2)$, output: $<(v_1, v_2), (v_1, v_2, u_1, u_2)>$ where u_1 and u_2 are generated by any 2-permutation of elements from U.

6.2

If our purpose is to find specific sized, e.g. t-, polygon, the algorithms should be modified as such:
Map input: $<, A>$ where u are map machine identified vertex and A is the adjacency matrix of input graph
Map output: $<, (k, (p_1, p_k), (p_2, p_3, ..., p_{k-1}))>$ where u is the vertex ID as the key. While for the value part, there are 2 tuples and a integer k. (p_1, p_k) stands for the start and the ending of a path, $(p_2, p_3, ..., p_(k-1))$ represents $k-1$ relay point on the path, and k is the number of points on the path. Also that $\forall i, j : i \neq j \Rightarrow p_i \neq p_j$

Reduce input: $<(p_1, p_k), (k, (p_2, p_3, ..., p_{k-1}))>$
Reduce output: $<(p_1, p_k), (p_1, p_2, ..., p_t)>$
Inside every reduce machine, the path pair P_i, P_j whose length $k_i + k_j = t - 2$ are stitch together, before output, every stitched polygon should be verified that the vertex in their vertex set $(p_1, p_2, ..., p_t)$ are mutual different. Each time the algorithm gives out a polygon, could we add $\frac{1}{t}$ to the count of polygons on all of the vertexes involved.

6.3

Bottle neck of this algorithm should be:

1. The path-finding procedure in map phase, it is with complexity of $\mathcal{O}(d^k)$ where d is the degree of a vertex and k is the length of paths be found.
2. The calculation of mutual different path pairs.
3. Every t-polygon are calculated t times using this algorithm.

Those problems could be partly solved by using the heavy hitter technique mentioned in the lecture, the all-combination of t vertexes whose degree is larger than some threshold $\tau = f(|E|)$ are at first calculated and verified. Or a technique of more-than-one level of Map-Reduce are also helpful by stitching the paths. For example, the first stage of M-R generates paths with length of 2 (with 3 vertexes), the second stage of M-R can then generate all paths with length of 4. With the modification of the length of path generated after each stage, the complexity of ring-generation can be reduced in ln-level.