Appendix B

Time and space complexity proofs of various algorithms are as follow.

1. Degree centrality

Degree centrality is a basic ranking algorithm for identifying node importance. The degree of node i is defined as follows:

$$k_i = \sum_{j=1}^{N} a_{ij}$$

(1) The calculation process of time complexity.

Traversing the network to obtain the degree of each node as its degree centrality value. Since the number of network nodes is n, the time complexity of this method is $O(n)$.

(2) The calculation process of space complexity.

Storing the degree centrality values of n nodes in the network, so the space complexity is $O(n)$.

2. K-shell decomposition method

The k-shell decomposition method proposed by Kitsak et al. Which is a graph-theoretical algorithm used to decompose an undirected graph into multiple k-shell subgraphs. A k-shell subgraph refers to a subgraph composed of nodes with degrees greater than or equal to k in the graph, that is, any node in the subgraph has a degree greater than or equal to k. The core idea of the algorithm is to repeatedly prune the nodes with the smallest degrees and update the degrees of their adjacent nodes until all nodes are decomposed into k-shell subgraphs.

(1) The calculation process of time complexity.

A. First traverse all nodes and sort their degrees, requiring $n + n\log n$ operations;

B. Iteratively delete nodes with degree 1 (at this time the k-shell value is 1), and continuously update the K-shell value from small to large, repeating this step until no nodes can be deleted. This operation does not seem to measure nodes well, but in fact, each node and its corresponding edges are deleted in turn, so it can be seen that as the K-shell value increases, a total of e edges are eventually deleted, so this step performs e operations;

In summary, the time complexity of the K-shell method is $O(n + n\log n + e)$.

(2) The calculation process of space complexity.

In the calculation process of the K-shell method, it is necessary to store the degree values of all n nodes in the network and the K-shell values of n nodes, requiring a total of 2n values. The space complexity is $O(2n)$.

3. Collective influence (CI)

The collective influence of a node is defined as follows:

$$CI_i = (k_i - 1) \sum_{j \in set(i,l)} (k_j - 1)$$

(1) The calculation process of time complexity.

Taking a step length of 2 as an example, first calculate the degree of each node, requiring n steps. Then calculate the number of all nodes within two steps of each node, so an average of $< k >^2$ nodes need to be explored. Therefore, a total of $n + n^* < k >^2 = n + \frac{e^2}{n}$ steps are required, and the time complexity is

$O(n+\frac{e^2}{n})$, where: $< k >= \frac{e}{n}$.

(2) The calculation process of space complexity.

Taking a step length of 2 as an example, since each node needs to record two types of information, one is the degree value of each node requiring storage of n values, and the second is the sequence number of each node and its adjacent nodes within 2 step lengths, requiring an average of $n* < k >^2 = \frac{e^2}{n}$ data storage.

Therefore, the space complexity is $O(n+\frac{e^2}{n})$.

4. Betweenness centrality (BC)

Betweenness centrality refers to the number of times a node appears in all shortest paths in a graph, that is, the importance of the node in connecting other nodes in the network. It is defined as follows:

$$BC_i = \sum_{i \in d(i,j)_{\min}, i \neq j} 1$$

(1) The calculation process of time complexity.

Since this method needs to explore the shortest path between every two nodes, the number of constructions is slightly different. Let's consider the worst case, for example, when all nodes form a large circular ring. Then, distinguish the number of nodes n as odd and even. The average path sum d of the ring structure is:

When n is odd, $d = 1+2+3...+\frac{n-1}{2}+\frac{n-1}{2}+...+2+1 = \frac{n^2-1}{8}$,

When n is even, $d = 1+2+3...+\frac{n}{2}-1+\frac{n}{2}+\frac{n}{2}-1+...+2+1 = \frac{n^2+2n}{8}$,

Since each node needs to perform the above operations, the total number of steps to operate a network is:

When n is odd, $\frac{n^3+2n^2}{8}$ steps are required,

When n is even, $\frac{n^3-n}{8}$ steps are required.

Therefore, regardless of whether n is odd or even, the time complexity required to calculate the entire network is: $O(n^3)$.

In addition, since the BC method needs to find the number of times each node participates in the shortest path, the ratio of the total number of edges e in the network to the average path length d/n of the network represents the average number of shortest paths in which each node participates.

When n is odd, $\frac{e}{d/n} = \frac{e}{\frac{n^2-1}{8n}} = \frac{8ne}{n^2-1}$,

When n is even, $\frac{e}{d/n} = \frac{e}{\frac{n+2}{8}} = \frac{8e}{n+2}$.

Therefore, the total number of times all nodes in the entire network are traversed is:

When n is odd, $\frac{8n^2 e}{n^2-1}$,

When n is even, $\frac{8en}{n+2}$.

In summary, the total number of operations in the odd and even cases are respectively:

When n is odd,

$$\frac{n^3+2n^2}{8}+\frac{8ne}{n^2-1}=\frac{n^5+2n^4-n^3+2n^3-2n+64en}{8n^2-8}=\frac{n^5+2n^4-n^3+2n^3-2n+64<k>n^2}{8n^2-8}\approx O(n^3)$$

When n is even,

$$\frac{n^3-n}{8}+\frac{8en}{n+2}=\frac{n^4-n^2+2n^3-2n+64en}{8n+16}=\frac{n^4-n^2+2n^3-2n+64<k>n^2}{8n+16}\approx O(n^3)$$

(2) The calculation process of space complexity.

The number of edge combinations that can be formed between any two points is $\frac{n(n-1)}{2}$.

When n is odd, the average path length of the network is: $\frac{n^2-1}{8n}$, so the average amount of information that needs to be stored is: $\frac{n^2-1}{8n} * n * (n-1)$, so the space complexity is: $O(n^3)$.

When n is even, the average path length of the network is: $\frac{n+2}{8}$, so the average amount of information that needs to be stored is: $\frac{n+2}{8} * n * (n-1)$, so the space complexity is: $O(n^3)$.

5. WL algorithm

The WL algorithm is a ranking method based on node degree and adjacent node degree. It is defined as follows:

$$WL_i = \sum_{j \in N(i)} \left(k_i \times k_j \right)$$

(1) The calculation process of time complexity.

The time complexity of this method is similar to the CI method. First, traversing the degree of each node in the entire network requires n steps. Then, for each node, you need to find its adjacent nodes, and each node has an average of $<k>$ adjacent nodes. Therefore, the WL method requires $n*<k>=e$ steps, so the time complexity is $O(e+n)$.

(2) The calculation process of space complexity.

traversing the degree values of nodes requires n steps; then calculating the degree values of adjacent nodes for each node requires an average of $<k>$ values to be stored, so a total of $n*<k>=e$ space is

required, so the space complexity is $O(e+n)$.

6. DWT algorithm

The DWT algorithm is a decomposition method proposed by YIRUN RUAN et al. It quantifies the strength of links based on local information of network topology and designs a simple and effective method to evaluate the importance of nodes according to the number of connections and overlap of neighbors.

$$DWT_i = \sum_{j \in N(i)} \left(1/k_i + S_{ij}/k_i \right)^2, S_{ij} = \frac{\left| N(i) \cap N(j) \right|}{\sqrt{k_i \times k_j}}$$

(1) The calculation process of time complexity.

First, calculating the degree of each node requires n steps; secondly, calculating the adjacent nodes of each node requires $n^* < k >= e$ steps, and performing the "and" operation between the neighbor nodes of each node itself and the neighbor nodes of the neighbor nodes one step away requires $n^* < k >^2 = \frac{e^2}{n}$ steps, so the DWT method cumulatively requires $n + e + n^* < k >^2 = n + e + \frac{e^2}{n}$ steps. Therefore, the time complexity is $O(n + e + \frac{e^2}{n})$.

(2) The calculation process of space complexity.

First, calculating the degree of each node requires recording n values. Then the number of adjacent node sets for each node is $n^* < k >= e$. Then the "and" operation between the adjacent node sets of each point and the adjacent node sets of its neighbor nodes yields $n^* < k >= e$ Solton values. Therefore, the final space complexity is $O(n+2e)$.

7. HC algorithm

The harmonic centrality (HC) method is defined as follows:

$$HC_i = \sum_{j \neq i} \frac{1}{d(i,j)}$$

(1) The calculation process of time complexity.

Since this method needs to explore the shortest path between every two nodes, the construction method of the shortest path is similar to the shortest path analysis of the BC method. Similarly, let's consider the worst case, for example, when all nodes form a large circular ring, when N is odd and even respectively, the sum of the average paths of the ring structure is:

When n is odd, $d = 1 + 2 + 3 \ldots + \frac{n-1}{2} + \frac{n-1}{2} + \ldots + 2 + 1 = \frac{n^2 - 1}{8}$,

When n is even, $d = 1 + 2 + 3 \ldots + \frac{n}{2} - 1 + \frac{n}{2} + \frac{n}{2} - 1 + \ldots + 2 + 1 = \frac{n^2 + 2n}{8}$,

Since each node needs to perform the above operations, the total number of steps to operate a network is:

When n is odd, $\frac{n^3 + 2n^2}{8}$ steps are required,

When n is even, $\frac{n^3-n}{8}$ steps are required.

Therefore, regardless of whether n is odd or even, the time complexity required to calculate the entire network is: $O(n^3)$.

(2) The calculation process of space complexity.

First, the number of edge combinations that can be formed between any two points is $n(n-1)$. Secondly, when n is even, the average path length of the network is: $\frac{n+2}{8}$ (that is, we need to store the information of $\frac{n+2}{8}$ edges), so the average amount of information that needs to be stored is $\frac{n+2}{8} * n * (n-1)$, so the space complexity is $O(n^3)$.

8. Closeness centrality

(1) The calculation process of time complexity.

Since the CC method also needs to calculate the shortest path, the time complexity calculation process is the same as above: $O(n^3)$

(2) The calculation process of space complexity.

Similarly, the space complexity of the CC method is: $O(n^3)$.

9. Pagerank

PageRank is an algorithm proposed by Page and Brin, the co-founders of Google, to rank web pages. The specific formula is as follows:

$$PR(i) = \frac{(1-d)}{n} + d * \sum_{j \in \Gamma(i)} \frac{PR(j)}{C(j)}$$

(1) The calculation process of time complexity.

First, initialize the PR value, which requires n steps. Then calculate the outdegree value of the adjacent nodes of each point, which requires $n*<k>=e$ steps. Then calculate the product with the damping factor, which requires n steps. Therefore, in summary, $2n+n*<k>=2n+e$ steps are required in total, so the time complexity is $O(2n+e)$.

(2) The calculation process of space complexity.

First, save the initial PageRank values of each node, requiring n initial values. Then record the outdegree value of each node's neighbor node, saving $n*<k>=e$ values. Then calculate the product with the damping factor, requiring n values. Therefore, a total of $2n+n*<k>=2n+e$ storage space is required, so the space complexity is $O(2n+e)$.

10. INCC

The INCC method combines the direct and indirect influences of a node's nearest neighbors and next nearest neighbors. The specific formula is as follows:

$$INCC_i = \sum_{j \in \Gamma(i)} \left(p'_{ij} + \sum_{k=1, k \neq i, j}^{n} p'_{ik} p'_{kj} \right)^2$$

(1) The calculation process of time complexity.

It takes n steps to calculate the degree of each node. Then it takes $n*<k>^2$ steps to calculate the sum of the degrees of the two-step neighbors N_w of each node. It takes $n*<k>^3$ steps to calculate Q_j. It takes $n*<k>^4$ steps to calculate LR_i. It takes n steps to calculate p'_{ij}. Then it takes $n<k>(1+<k>^2+1)=e(2+<k>^2)$ steps to calculate the $INCC_i=\sum_{j\in\Gamma(i)}\left(p'_{ij}+\sum_{k=1,k\neq i,j}^{n}p'_{ik}p'_{kj}\right)^2$ of each point. In summary, $n+\frac{e^2}{n}+2\frac{e^3}{n^2}+\frac{e^4}{n^3}+2e$ steps are required. Therefore, the time complexity is $O(n+\frac{e^2}{n}+2\frac{e^3}{n^2}+\frac{e^4}{n^3}+2e)$.

(2) The calculation process of space complexity.

First, the degree value of each node needs to be stored, requiring n data. Then n N_w need to be stored, further n Q_j values need to be stored, n LR_i values need to be stored, and finally n $INCC_i=\sum_{j\in\Gamma(i)}\left(p'_{ij}+\sum_{k=1,k\neq i,j}^{n}p'_{ik}p'_{kj}\right)^2$ values need to be stored after calculating the INCC value. In total, $5n$ data need to be stored. In summary, the space complexity is $O(5n)$.

11. KPDN

Please refer to the paper for details.

(1) The calculation process of time complexity.

A. Since our algorithm first uses the improved k-shell method, it is first necessary to traverse and sort n nodes, requiring $n+n\log n$ operations; secondly, as the k-shell value increases, a total of e edges are deleted, so a total of $n+n\log n+e$ steps are required to complete the improved k-shell operation;

B. Then calculate the Solton index of the adjacent nodes of each node. First, calculating the adjacent nodes of each node requires $n*<k>=e$ steps. Then performing the "and" operation between the neighbor node set of each node and the neighbor node set of its neighbor nodes requires $n*<k>^2=\frac{e^2}{n}$ steps;

C. Then perform addition and multiplication operations on each node, requiring n steps; Therefore, in summary, a total of $n+n\log n+e+\frac{e^2}{n}+n=2n+n\log n+e+\frac{e^2}{n}$ steps are required, so the time complexity is $O(2n+n\log n+e+\frac{e^2}{n})$.

(2) The calculation process of space complexity.

First, it is necessary to store the k-shell values, degree values and the order values of each node peeled off

under the same k-shell value of n nodes, so $3n$ data volume is required here. Secondly, each node needs to calculate the Solton index, so $n^* < k >$ values need to be stored. Finally, n values are obtained by addition and multiplication, so a total of $4n + n^* < k >= 4n + e$ values need to be stored, and the space complexity is $O(4n + e)$.

Note: It should be noted here that the complexity in our draft calculation has not been simplified, because we do not know the actual network situation in the actual use process. For example, when <k> is small in the network, we can set e=n*<k>. Therefore, the above complexity can be simplified to:

Methods	**Time complexity**	**Space complexity**
DC	$O(n)$	$O(n)$
K-shell	$O(n + n\log n + e)$	$O(2n)$
CI	$O(n + \frac{e^2}{n})$	$O(n + \frac{e^2}{n})$
BC	$O(n^3)$	$O(n^3)$
WL	$O(e + n)$	$O(e + n)$
DWT	$O(n + e + \frac{e^2}{n})$	$O(n + 2e)$
HC	$O(n^3)$	$O(n^3)$
CC	$O(n^3)$	$O(n^3)$
Pagerank	$O(2n + e)$	$O(2n + e)$
INCC	$O(n + \frac{e^2}{n} + 2\frac{e^3}{n^2} + \frac{e^4}{n^3} + 2e)$	$O(5n)$
KPD	$O(n + n\log n + e)$	$O(3n)$
KPDN	$O(2n + n\log n + e + \frac{e^2}{n})$	$O(4n + e)$