# 편심 기반 전략 네트워크

연구자: Melese Medhin(medhinmele10@gmail.com)
책임지도자: Jerome Tambour(수리정보과학부, 한국과학영재학교, jerome.tambour@gmail.com)

# Eccentricity-based strategic network

Researcher: Melese Medhin(medhinmele10@gmail.com)
Supervisor: Jerome Tambour(Department of Mathematics and Computer Science, KSA OF KAIST, jerome.tambour@gmail.com)

**국문초록**

전략적 네트워크는 관계를 유지하거나 원하지 않는 개인의 선택에 의해 형성된 사회적 및 경제적 네트워크 중하나입니다. 개인적 인센티브를 배양하거나 전반적인 사회 복지를 극대화하기 위해 형성 될 수 있습니다. 후자의 경우 효율성은 가장 큰 성과를내는 네트워크를 식별하는 데 도움이되는 중요한 개념입니다. 이 백서에서는 특정 조건에서 편심 기반 전략 네트워크에서 효율적인 네트워크 유형을 조사합니다. 조건에 대해 이야기 할 때 링크를 유지하기 위해 집불해야하는 상대적인 비용과 링크에서 누리는 혜택에 대해 이야기하고 있습니다. 이 질문에 답하기 위해 우리는 주어진 개인의 보수와 네트워크의 총 보수를 계산할 수있는 효용 함수의 개념을 사용했습니다. 주어진 개인의 보수는 학위, 비용 및 혜택 함수의 세 가지에 달려 있습니다. 이익 함수는 편심 함수이며 감소 함수입니다. 그 후 몇 가지 유형의 중요한 네트워크를 고려하고 각각에 대한 효용 함수 공식을 도출하고 주어진 조건에서 총 효용을 서로 비교했습니다.

**Abstract**

Strategic network is one of social and economic networks that is formed by the choice of individuals who want to maintain or not a relationship. It can be formed either to cultivate personal incentives or to maximize the overall societal welfare. In the case of the latter, efficiency is an important concept that helps us to identify a network with the greatest payoff. This paper examines which types of network are efficient in eccentricity-based strategic network under certain conditions. When talking about conditions, we are talking about the relative amount of a cost has to be paid to maintain a link and a benefit that is enjoyed from the link. In order to answer this question, we used the concept of utility function which enables us to calculate the payoff of a given individual as well as the total

payoff a network. The payoff of a given individual depends on three things: those are a degree, a cost and a benefit function. The benefit function is a function of eccentricity and it is a decreasing function. After that, we considered some types of important networks, derived a utility function formula for each of them and compared their total utility with each other under given conditions

**Key words: strategic network, graph theory, network efficiency, eccentricity**

## I. Introduction

Social and economic networks are the foundation of every society. They are important in our daily lives in order to obtain information that enable us to make several important decisions. Not only information, but also many other benefits might come through network; the individuals in a given network can access material benefits too. These benefits don't come freely, but at the expense of some cost to maintain a link (relationship). On one hand, having many links might bring greater benefit; on the other hand, it might require greater price. The concept of strategic network analyzes this trade-off between benefit and cost.

In this report, we will study strategic networks, that is how people decide to make a connection based on cost and benefit. We are motivated by the distance-based utility model introduced by Jackson in [1] and introduced our own model, which is called eccentricity-based strategic network. There are some similarities and differences between these two models. In both cases, the utility of an individual in a given network is dependent on a decreasing benefit function, cost and number of neighborhoods. In distance-based model an individual can obtain different amounts of benefit from its different neighbors located at a given distance, and the benefits are summed up. And also the utility from direct links and indirect links are treated differently; an individual can benefit from its indirect links without paying any cost. But, in our model there is only single benefit that an individual gets depending the value of its eccentricity.

Our main target is to identify the most efficient networks under given conditions. To do this, first we have to consider different types of networks, then derive the total utility of each of them and finally compare their total utility to find the efficient network.

This paper is divided in to three sections. In the first section, we briefly discuss the concepts related to strategic networks and the distance-based utility model from preliminary studies. In particular, definitions and basic notations, which help to understand our work are introduced. In the second section, we explain our model of strategic network; the concept of eccentricity-based strategic network is discussed. Formulas of utility function for different networks, propositions and their proofs are included too. In third and final section, we conclude our work.

## II. Theoretical Background

### A) Networks

Networks are represented by graphs, made of nodes related by links to represents a relationship between the nodes. The concept of node includes people, firms, organization and basically any set of things in a relationship. Although there are many methods to describe graphs, we preferred using the concept of adjacency matrix. Definitions and concepts in this report are from [1].

**Definition 1:** The Set N = {1, . . ., n} is the set of nodes that are involved in the relationship.

**Definition 2:** A graph (N,g) consists of

1. A set N = {1,..., n} of nodes (or vertices, or agents) and

2. An adjacency matrix g of dimension n in which $g_{ij}$ is equal to 1 if there is a link between node i and node j, and 0 if there is no link.

A network can be either undirected graph or directed graph. In case of undirected networks, two nodes are either connected or not. Facebook friendship is a typical example. For directed network, the first node can be connected to the second node without the second being connected to the first, the one like in Instagram. Throughout the whole report, only undirected networks are considered. Also, networks have no loop (gii=0)

**Definition 3:** A subgraph (N',g') of a graph (N,g) is a graph whose nodes and links are subsets of the graph (N,g).

### 1) Paths and cycles

**Definition 4:** A walk between node i and j in a network g is a sequence of links $i_1i_2$, $i_2i_3$, …,

$i_{T-1}i_T$ such that $i_ti_{t+1}$ is a link in network g for every $t\in\{1, \ldots, T-1\}$, and $i_1 = i$ and $i_T=j$. It is a path when $i_1,i_2 \ldots i_T$ are distinct nodes. Moreover,

- A cycle is a walk that starts and ends at the same node.
- The distance between two nodes is the length of the shortest path between them.

Consider the adjacency matrix $g = \begin{pmatrix} 0&1&1&0 \\ 1&0&0&1 \\ 1&0&0&1 \\ 0&1&1&0 \end{pmatrix}$ for example. The entry at row i and column j of the matrix $g^2 = \begin{pmatrix} 2&0&0&2 \\ 0&2&2&0 \\ 0&2&2&0 \\ 2&0&0&2 \end{pmatrix}$ is the number of walks of length of 2 between any two nodes i and j. More generally, the number of walks of length k between nodes i and j is given by the entry at row i and column j of matrix $g^k$.

### 2) Component graph and connected graph

**Definition 5:** We say a graph is connected if there is a path from every node to every other node.

**Definition 6:** A subgraph (N',g') of a graph (N,g) is a component if g' is connected and there are no edges in g between a node in N' and a node in N-N'.

**Definition 7:** Let $G_1$, $G_2$,…, and $G_r$ be r connected graphs (including isolated points). We denoted by $G=G_1+G_2+\ldots+G_r$ the graph whose components are $G_1$, $G_2$,…, and $G_r$

3) Degree of a node in network g

**Definition 8:** The degree of a node i is the number of links that involve the node i and it is denoted as $d_i(g)$.

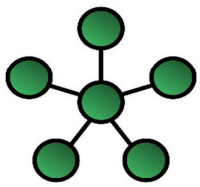
4) Eccentricity of a node

**Definition 9:** The eccentricity of a node i in a network g is the maximum of distances from i to any other node. The eccentricity of a node i in the network g is represented by $e_i(g)$.

B) Important examples of network

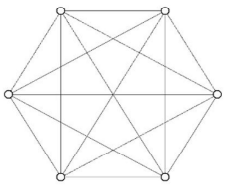
Important networks that we investigate in this report are star, empty, complete and double-node networks. Let's discuss each of them turn by turn.

- A Tree is a connected network that has no cycles.
- Star network is a tree network with one node that is connected to all other nodes. The star network $S_n$ has n nodes and exactly n-1 links.



[Fig 1] star network with 6 nodes and 5 links, from [2]

- The empty network with n nodes is the network that has no edges between any two nodes. It is represented by $E_n$.
- The complete network with n node is the network in which has an edge between every pair of different nodes. It is represented by $K_n$.



[Fig2] A complete network with 6 nodes; i.e. $K_6$, from [3]

- The double-node network with n even nodes is the network in which every node is paired and if we have n odd nodes every node is paired except one node which exists alone. It is represented by $D_n$.

[Fig3] A double-node network with 4 nodes

[Fig4] A double-node network with 5 nodes

C) Strategic networks

Strategic networks are formed based on the costs and benefits. The consent of players plays a major role in its formation. The players are provided with the right to cut a link if it isn't beneficial to them. Moreover, a link is formed between two players by considering mutual benefit. The strategic network is defined by **a utility function** that governs the overall benefit net of costs that a player enjoys in a given network. The utility of node i in the network g is represented by $U_i(g)$.

**Definition 10:** The network that has the largest total utility among all networks with n nodes is called **efficient**

The concept of **efficiency** addresses the total social welfare of a network. It mainly focuses on the efficiency of the network as whole, thus it enables to determine the network that gives the maximum total utility. The **total utility** is defined as the sum $U(g) = \sum_{i=1}^{n} U_i(g)$ of individual utilities.

1) Distance-based utility model

In the distance-based utility model, players get benefit from their direct connections as well as from indirect connections. But only direct connections are costly. For indirect connections, the utility deteriorates with the increasing distance between players.

Consider this: let b be a decreasing function defined on the set of positive integers, and c a positive real number.

In a given network g, a link has a cost c for each of its node. Moreover, each node i gets a benefit of $b(k_{ij})$, where $k_{ij}$ is

distance between two nodes, i and j, from every node j in the same component. Then the utility U of a given player in the network g can be written as

$$U_i(g) = \sum_{i \neq j} (\mathbf{b(kij)} - \mathbf{di(g)c})$$

where the sum runs over all vertices j that are in the same component as i.

In this model adding links provides benefits by reducing the distance between nodes, but there is also a cost for the added links. So there is a trade-off during the addition of links.

Jackson (see proposition 6.3.1 in [1]) shows that a complete, an empty and a star graph can be uniquely efficient in consideration of benefit and cost.

Theorem(Jackson[1]): In the distance-based model, the unique efficient network is

(i) the complete network if $b(2) < b(1) - c$,

(ii) a star encompassing all nodes if $b(1) - b(2) < c < b(1) + (n-2)/2 * b(2)$)and

(iii) the empty network if $b(1) + (n-2)/2 * b(2) < c$.

## III. Research Results

### 1) Eccentricity-based utility model

In this model, the utility of a given node is computed using the benefit function, cost and degree of a node. Let b be a decreasing function defined on the set of positive integers. Then the benefit for the node i is $b(e_i(g))$ where $e_i(g)$ is its eccentricity; and the cost for each link involving the node i is a positive real number c. Therefore, the eccentricity-based utility of a given node i in the network g can be written as

$$U_i(g) = b_{(ei(g))} - d_i(g)c$$

where c the cost for direct links and $d_i(g)$ is the degree of i in the network g.

The **total utility** of the network is defined as the sum $U(g) = \sum_i^n U_i(g)$ of individual utilities. We now know the general formula to find the utility of a node as well as the total utility in our model. But, these formula takes different shapes under different set of networks such as star, complete and so on. So, what is the total utility of empty network, star network, complete network and double-node network?

**Proposition 1**: $U(K_n) = n[b_1 - c(n-1)]$

Proof: Every node in the complete network has an eccentricity of 1 and a degree of n-1. Then, the utility of one node is $b_1 - c(n-1)$. Since there are n nodes, $U(K_n) = n[b_1 - c(n-1)]$

**Proposition 2**: $U(E_n) = 0$

Proof: In an empty network, nodes neither gain a benefit nor pay for the cost. So, its total utility is 0.

**Proposition 3**: $U(S_n) = b_1 + (n-1)(b_2 - 2c)$ for $n \geq 3$

Proof: Assume that $n \geq 3$

The utility of a central node is $b_1 - c(n-1)$. The utility of each of the peripheral node is $b_2 - c$; there are n-1 peripheral nodes. So the total utility of all peripheral nodes is $(n-1)(b_2 - c)$.

The total utility of star network with n node is the sum of the utility of a central node and the utility of all peripheral nodes; that is, $U(S_n) = b_1 + (n-1)(b_2 - 2c)$.

**Proposition 5**: $U(D_n) = n(b_1 - c)$ if n is even and $U(D_n) = (n-1)(b_1 - c)$ if n is odd.

Proof: When n is even, there are n nodes in which each of them has a utility of $b_1 - c$. So, $U(D_n) = n(b_1 - c)$. When n is odd, there are n-1 nodes in which each of them has a utility of $b_1 - c$ and one isolated node. So,

$U(D_n) = (n-1)(b_1 - c)$ and $U(D_n) > 0 \quad \Leftrightarrow \quad c < b_1$

**Proposition 6**: If $c > b_1$, then the empty network is the only the efficient one.

Proof: Assume that g is a connected network. Then the utility of node i in g is

$$U_i(g) = b_{(ei(g))} - d_i(g)c$$

We know that $b_{(ei(g))} \leq b_1$ and $d_i(g)c \geq c$. So, if $c > b_1$ then $U_i(g) < 0$. Therefore, $U(g) < U(E_n) = 0$.

When g is not connected, it means that g has either component(s) without isolated node(s) or component(s) with isolated node(s). The total utility of each component is negative as shown in the above proof, and isolated node (s) has a utility of 0. So, $U(g) < 0 = U(E_n)$. Therefore, the empty network is the only efficient one for $c > b_1$.

**Propostion7**: If $n \geq 3$ and $m \geq 3$, then $U(S_{n+m}) < U(S_m + S_n)$

Proof: Assume that $n \geq 3$ and $m \geq 3$. We have $U(S_{n+m}) = b_1 + (m+n-1)(b_2 - 2c)$, and $U(S_m + S_n) = 2b_1 + (m+n-2)(b_2 - 2c)$.

So, $U(S_m + S_n) - U(S_{n+m}) = b_1 + b_2 + 2c > 0 \quad \Leftrightarrow \quad c > (b_2 - b_1)/2$

Since c is a positive real number, it is always greater than $(b_2 - b_1)/2$. Therefore, $U(S_{n+m}) < U(S_m + S_n)$.

**Proposition 8**: If $n \geq 3$, then $U(S_{n+2}) < U(S_n + S_2)$

Proof: Assume that $n \geq 3$. We have $U(S_{n+2}) = b_1 + (n+1)(b_2 - 2c)$, and $U(S_n + S_2) = 3b_1 + (n-1)(b_2 - 2c) - 2c$.

So , $U(S_n + S_2) - U(S_{n+2}) = 2(b_1 - b_2 + c) > 0 \Leftrightarrow c > b_2 - b_1$

Since c is a positive real number, it is always greater than $b_2 - b_1$. Therefore, $U(S_{n+2}) < U(S_n + S_2)$.

**Proposition 9**: $U(S_4) < U(S_2 + S_2)$

Proof: we have $U(S_2 + S_2) = 4(b_1 - c)$ and $U(S_4) = b_1 + 3(b_2 - 2c)$ so , $U(S_2 + S_2) - U(S_4) = 3(b_1 - b_2) + 2c > 0$.

**Conclusion 1**: From proposition 7, proposition 8 and proposition 9, we can conclude that if $m \geq 2$ and $n \geq 2$ then $U(S_{m+n}) < U(S_m + S_n)$.

**Proposition 10**: For $n \geq 3$, $U(S_{n+1}) < U(S_n + S_1)$ if and only if $c > b_2/2$

Proof: Assume that $n \geq 3$. We have $U(S_{n+1}) = b_1 + n(b_2 - 2c)$, and $U(S_n + S_1) = b_1 + (n-1)(b_2 - 2c)$

So, $U(S_n + S_1) - U(S_{n+1}) = 2c - b_2 > 0 \quad \Leftrightarrow \quad c > b_2/2$

**Proposition 11**: $U(S_3) < U(S_2 + S_1)$ if and only if $c > (2b_2 - b_1)/2$

Proof: we have $U(S_2 + S_1) = 2(b_1 - c)$, and $U(S_3) = b_1 + 2(b_2 - 2c)$

So, $U(S_2 + S_1) - U(S_3) = b_1 - 2b_2 + 2c > 0 \quad \Leftrightarrow \quad c > (2b_2 - b_1)/2$

**Proposition 12**: $U(S_2) < U(S_1 + S_1)$ if and only if $b_1 < c$

Proof: we have $U(S_1 + S_1) = 0$, and $U(S_2) = 2(b_1 - c)$

So, $U(S_1 + S_1) - U(S_2) = -2(b_1 - c) > 0 \quad \Leftrightarrow \quad b_1 < c$

**Proposition 13**: $U(S_3) + U(S_1) < U(2S_2)$

Proof: we have $U(2S_2) = 4(b_1 - c)$ and $U(S_3 + S_1) = b_1 + 2(b_2 - 2c)$

So, $U(2S_2) - U(S_3 + S_1) = 3b_1 - 2b_2 > 0$

**Proposition 14**: $U(S_3 + S_3) < U(S_2 + S_2 + S_2)$

Proof: we have $U(S_2 + S_2 + S_2) = U(D_6) = 6(b_1 - c)$, and $U(S_3 + S_3) = 2b_1 + 4b_2 - 8c$

So, $U(S_2 + S_2 + S_2) - U(S_3 + S_3) = 4(b_1 - b_2) + 2c > 0$

**Proposition 15**: Assume that $c < b_1$. Then, among all networks whose components are all stars:

- If n is even, then the only efficient network is $D_n$.
- If is odd, then the only efficient network is $D_n$ if $c > (2b_2 - b_1)/2$ or $D_{n-3} + S_3$ if $c < (2b_2 - b_1)/2$

Proof: Assume that n is even and g is an efficient network.

Let's write $g = [s_{e_1} + s_{e_2} + s_{e_3} + \ldots + s_{e_k}] + [s_{o_1} + s_{o_2} + s_{o_3} + \ldots + s_{o_l}]$ where $e_1, e_2 \ldots e_k$ are even integers such that $e_1 \leq e_2 \leq \ldots \leq e_k$ and $o_1, o_2, \ldots o_l$ are odd integers such that $o_1 \leq o_2 \leq \ldots \leq o_l$ and $n = [e_1 + e_2 + \ldots + e_k] + [o_1 + o_2 + \ldots + o_l]$.

From **conclusion 1**, If $e_k \geq 4$, then $U(S_{e_k}) < U(S_{e_k - 2} + S_2)$. This contradicts with the assumption that g is an efficient

network. So, $e_k < 4$. Therefore, $e_1 = e_2 = \ldots = e_k = 2$. This means that every even star component has only two nodes.

From **conclusion 1,** If $o_l \geq 5$, then $U(S_{o_l}) < U(S_{o_l-2} + S_2)$. This contradicts with the assumption that g is an efficient. So, $o_l < 5$. Therefore, $o_l \in \{1,3\}$. This means that every odd star component has either 1 node or three nodes.

Let r be number of star components with 1 node and s be number of start components with 3 nodes. So $o_1+o_2+\ldots..+o_l = 1+1+\ldots..+1 + 3+3+\ldots..+3$.

From **proposition 14,** If $s \geq 2$, then $U(S_3+S_3) < U(S_2 + S_2 + S_2)$. This contradicts with the assumption that g is an efficient. So $s < 2$. Therefore, s=0 or s=1.

From **Proposition 12,** If $r \geq 2$, then $U(S_1 + S_1) < U(S_2)$. This contradicts with the assumption that g is an efficient. So $r<2$. Therefore, r=0 or r=1.

Since n is even we have two options; either r=s=0 or r=s=1

From **Proposition 13**, If r=s=1, then $U(S_3 + S_1) < U(S_2 + S_2)$. This contradicts with the assumption that g is an efficient. Therefore, r=s=0 and $g = D_n = S_2 + S_2 + \ldots\ldots + S_2$

Proof when n is odd: Assume that n is odd and g is an efficient

Let's write $g = [s_{e_1} + s_{e_2} + s_{e_3} + \ldots + s_{e_k}] + [s_{o_1} + s_{o_2} + s_{o_3} + \ldots + s_{o_l}]$ where $e_1, e_2 \ldots e_k$ is even such that $e_1 \leq e_2 \leq \ldots \leq e_k$ and $o_1, o_2, \ldots o_l$ is odd such that $o_1 \leq o_2 \leq \ldots \leq o_l$ and $n = [e_1+e_2+\ldots+e_k]+[o_1+o_2+\ldots+o_l]$

From **conclusion 1**, If $e_k \geq 4$, then $U(S_{e_k}) < U(S_{e_k-2} + S_2)$. This contradicts with the assumption that g is an efficient network. So, $e_k < 4$. Therefore, $e_1 = e_2 = \ldots = e_k = 2$. This means that every even star component has only two nodes.

From **conclusion 1,** If $o_l \geq 5$, then $U(S_{o_l}) < U(S_{o_l-2} + S_2)$. This contradicts with the assumption that g is an efficient. So, $o_l < 5$. Therefore, $o_l \in \{1,3\}$. This means that every odd star component has either 1 node or three nodes.

Let r be number of star components with 1 node and s be number of start components with 3 nodes. So, $o_1+o_2+\ldots..+o_l = 1+1+\ldots..+1 + 3+3+\ldots..+3$,

From **proposition 14,** If $s \geq 2$, then $U(S_3+S_3) < U(S_2 + S_2 + S_2)$. This contradicts with the assumption that g is an efficient. So, $s < 2$. Therefore, s=0 or s=1.

From **Proposition 12,** If $r \geq 2$, then $U(S_1 + S_1) < U(S_2)$. This contradicts with the assumption that g is an efficient. So, $r < 2$. Therefore, r=0 or r=1.

Since n is odd, we have two options: either r=1 and s=0 or r=0 and s=1

From **proposition 11**, if $c > (2b_2 - b_1)/2$, then $U(S_2 + S_1) > U(S_3)$. Therefore, r=1 and s=0 and $g = D_n = S_2 + S_2 + \ldots..+ S_2 + S_1$. From the same **proposition 11,** if $c < (2b_2 - b_1)/2$, then $U(S_3) > U(S_2 + S_1)$, so r =0 and s=1 and $g = D_{n-3} + S_3$.

### 2) Python programming

We used python to implement functions as it has been difficult to make calculations manually. By using it, we created star network, empty network and complete network and computed their utilities. Some of the important functions are:

- Power(G,k): returns $G^k$ for a square matrix G and an integer k
- Distance(G,i,j): returns the distance between node i and j in network G(returns -1 if i and j are not connected)
- Eccentricity(G,a): returns the maximum of distances from node a to other nodes in network G.

The python codes are shown in the **appendix**.

## IV. Conclusion

The total utility of different networks is computed in the eccentricity-based model. In particular, this concept is used to define the benefit function. The benefit decreases with increasing value of eccentricity. The overall payoff a network was calculated and the networks with the greatest utility are identified under certain conditions. We conjecture that the double-node network is the efficient type of network for even n and relatively small costs, whereas, the empty network is the most efficient for relatively large costs.

## ■ References

[1] Matthew O. Jackson, Social and Economic Networks, 2008

[2] https://en.wikipedia.org/wiki/Star_network

[3] https://upload.wikimedia.org/wikipedia/commons/d/d9/Complex_network_K6_complete_graph.png

## ■ Appendix

```
import numpy as np
import random as rand

G = np.array(
    [[0,0,0,0],
     [0,0,1,1],
     [0,1,0,0],
     [0,1,0,0]]
)


# degree of node a in network G

def degree (G,a):
    return sum(G[a])

# distance (by using power graph)
def power(G,k):
    if k==0:
        return np.identity(len(G))
    elif k==-1:
        return G
    else:
        return G.dot(power(G,k-1))



def distance(G,i,j):
    n = len(G)
    if i==j:
        return 0
    else:
        d=0
        k=1
        while k<=n-1 and d==0:
            if power(G,k)[i,j]>0:
                return k
            k = k+1
        return -1

# benefit function
def benefit(s,k):
     return s**k

# numner of nodes
def numberofnodes(G):
    return len(G)
```

```
# creating empty network
def create_Empty_Network(n):
    return np.array([[0]*n for _ in range(n)])

# creating star network
def create_Star_Network(n,center):
    G = [[0]*n for _ in range(n)]
    for i in range(n):
        if i!= center:
            G[center][i] = 1
            G[i][center] = 1

    return np.array(G)

# creating random network
def random_network(n,p):
    L = create_Empty_Network(n)
    for i in range(n):
        for j in range(i+1,n):
            r = rand.random()
            if r<p:
                L[i][j] =1
                L[j][i]=1
    return L

# eccentrcity of node i
def eccentricity(G,a):
    n = len(G)
    max = 0
    for i in range(n):
        d = distance(G,a,i)
        if i != a and d > max:
            max = d

    return max
```

```
# utility of node i in distance-based utility model
def utility(G,s,a,c):
    utility  = 0
    n = numberofnodes(G)
    for i in range(n):
        k = distance(G,a,i)

        if k>0:
            utility = utility + benefit(s,k)

    return utility - c*degree(G,a)

# total utility of a network in distance-based utility model

def totalutility(G,s,c):
    n = numberofnodes(G)
    sum = 0
    for i in range(n):
        sum += utility(G,s,i,c)

    return sum

# creating complete network
def create_Complete_Network(n):
    G = [[1]*n for _ in range(n)]
    for i in range(n):
        G[i][i] = 0

    return np.array(G)
```