Modeling the TikTok 'For You Page' Algorithm Using Graph Theory and Algorithm Complexity

Muhammad Fithra Rizki - 13523049[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13523049@std.stei.itb.ac.id

***Abstract*—TikTok has become one of the most popular social media platforms in the world. TikTok itself has a variety of very interesting features so that over time more and more internet users are interested in this social media. TikTok has a variety of interactions that are very user-friendly. They prioritize the comfort of their users so that they are designed in such a way in their application to make users immersed in using their application. TikTok itself has a very interesting feature, namely For Your Page. FYP itself can adjust users from interactions like, comment, share, save, and watch time users on a topic. This FYP system can be created with a graph theory approach and time complexity calculations so that it can be found how this FYP system works.**

***Keywords*—TikTok, FYP, Graph, Time Complexity**

I. Introduction

TikTok has become a social media application that is very often used by internet users. If TikTok is seen from the development process, TikTok is an application made by a company called Bytedance from China. Bytedance itself initially created an application called Douyin in 2016 which could only be used for internet users in China. The use of content on Douyin is basically the same as TikTok, which contains a collection of short videos. Then, in 2017, Bytedance made changes to this application so that this application became TikTok which could be used outside China. Then, in the same year, Bytedance bought an application called Musical.ly which contained the same content and was viral at that time. Then, Bytedance merged the two applications and developed them from 2018 until they became the TikTok application that we know today.

TikTok is increasingly increasing its features. Initially, TikTok was only used as a means of entertainment in the form of short videos that were relevant to its users. However, over time, TikTok can be used as a means of live streaming to an online shop called TikTok Shop. Not a few people have made a profit by becoming a TikTok creator.

The interesting thing to talk about from Tiktok is related to the algorithm in the For Your Page section. For Your Page is designed by Tiktok in such a way that it finds a point where users see videos that are relevant to watch. In addition to adjusting user interactions, Tiktok also looks at the achievement of a video's interaction such as views, likes, comments, shares, and saves. Regarding how Tiktok processes data and interactions from its videos, it is still a mystery for Tiktok.

However, the For Your Page Tiktok system can be designed simply using various theories, one of which is graph theory and assisted by calculations of algorithm complexity. Therefore, in this study, the researcher used the approach of both materials, namely graph theory and algorithm complexity, so that it is possible to create a simple system from For Your Page on Tiktok.

II. Theoretical Basis

A. TikTok

TikTok is an application used as a medium of entertainment for internet users. TikTok offers various things, especially in the form of short entertainment videos. TikTok has a fairly interesting algorithm so that what is displayed is based on user interaction with the types of content that are usually watched or items that are usually purchased by users.

In its use, TikTok has several main features to provide the best experience to its users, including:

1. For Your Page
   For Your Page on TikTok provides users with the best experience. This is because the For Your Page page uses the results of user interaction analysis as its main axis. Not only that, viral videos are also a determining factor for the existence of this For Your Page. So, on the For Your Page page, short videos that pass through the user's device have direct relevance from the analysis of what users often see to what users like.
2. User Interaction
   There are various interactions that users can do when viewing or watching a video, including like, comment, share, and save. This is one of the determinants in measuring the For Your Page algorithm. When a user performs these interactions, the possibility of the video appearing again on For Your Page becomes higher. So, the For Your Page algorithm really adjusts what is interesting to the user.
3. Duet and Stitch
   Duet and stitch are two features that users can do as special interactions to a video. Duet is a feature that allows users to respond directly through a video in a

side-by-side format with the duet video. As for stitch, the user's response video will be displayed after the original video is partially displayed.

4. Live
   Users can do or watch live streaming on TikTok. This feature allows users to create or watch an activity in real time. Currently, the live feature is most often used for selling, casual chatting, and others. This feature also allows live creators to interact directly with users who watch their live.
5. TikTok Shop
   TikTok shop is one of the newest features on TikTok. The TikTok shop system is the same as other online shops that allow users to shop directly on TikTok. Usually this feature can be directly connected to the creator who is live or videos from the creator.

B. Graph

A graph is a mathematical theory that broadly explains the relationship between two or more objects. A graph has two main components, namely nodes (points) connected by edges (lines). Graphs are commonly used in problems related to determining problems between entities and have been used in various fields, such as computer science, biology, and others.

Graphs have several forms and types based on the edge, namely as follows:

1. Simple Graph
   A simple graph is a graph that does not have ring edges or edges connected from a vertex to itself. This graph also does not have more than one edge connected to the same vertex or is called a multiple edge.

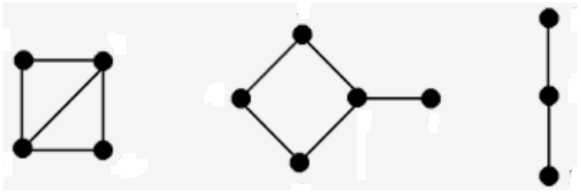


Fig 2.1 Simple graph
(Source : Graph (Part 1) Slide by Dr. Rinaldi Munir)

2. Unsimple Graph
   A simple graph is a graph that has ring edges or edges connected from a vertex to itself. This graph also has more than one edge connected to the same vertex or is called a multiple edge.

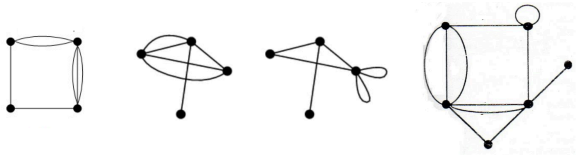


Fig 2.2 Unsimple graph
(Source : Graph (Part 1) Slide by Dr. Rinaldi Munir)

Unsimple graphs are divided into several types, as follows:

a. Multi Graph, a graph that contains multiple edges or has more than one edge connected from one vertex to another.
b. Pseudo-Graph, a graph that contains ring edges or contains vertices that have an edge connected to the vertex itself

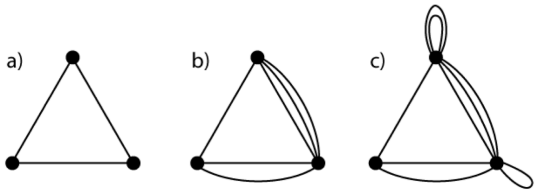


Fig 2.3 a) Simple graph, b) Multi graph, c) Pseudo-graph
(Source : Graph (Part 1) Slide by Dr. Rinaldi Munir)

Graphs can also be classified from the direction of their sides, as follows:

1. Undirected Graph
   Undirected Graph means that the graph does not have directional arrows between each other node on its side.

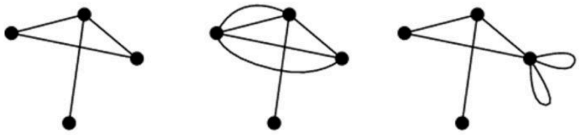


Fig 2.4 Undirected Graph
(Source : Graph (Part 1) Slide by Dr. Rinaldi Munir)

2. Directed Graph (Digraph)
   A directed graph is a graph that has a direction orientation on the edges between its vertices.

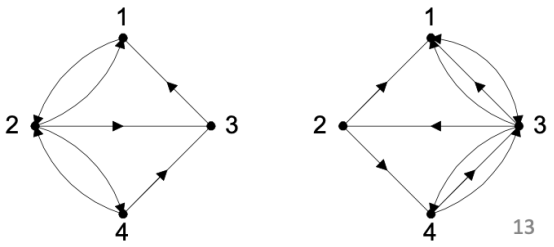


Fig 2.5 Directed graph
(Source : Graph (Part 1) Slide by Dr. Rinaldi Munir)

Graphs also have types where the graph has numbers that describe the weight on an edge. This value can represent distance, cost, or other parameters. This graph is called a weighted graph.

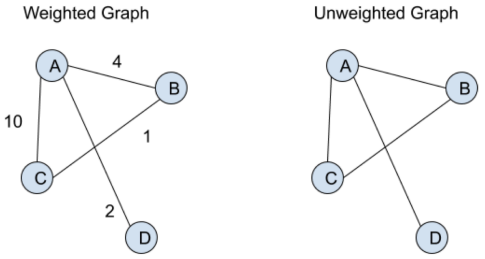


Fig 2.6 Weighted graph
(Source : Graph (Part 1) Slide by Dr. Rinaldi Munir)

There is also a graph called a bipartite graph. This graph has vertices that are separated into two parts and the edges that exist connect the vertices from the two different parts.

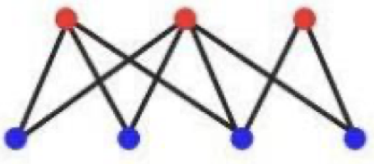


Fig 2.7 Bipartite graph
(Source : Graph (Part 3) Slide by Dr. Rinaldi Munir)

Graphs can be classified based on the intersection of their edges. If a graph does not have at least one intersection of two edges, then the graph is called a planar graph, if there is an intersection, the graph is called a non-planar graph. A planar graph that is described in another way that does not have an intersection is called a plane graph.

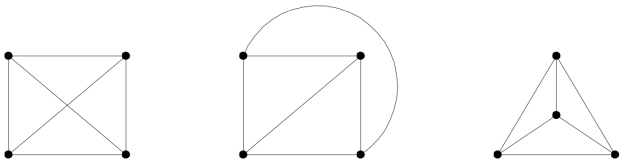
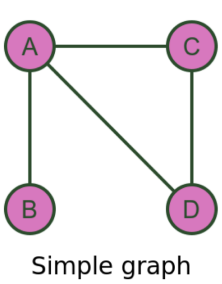


Fig 2.8 Planar graph, b) and c) also called as plane graph
(Source : Graph (Part 2) Slide by Dr. Rinaldi Munir)

Graphs can be represented in various ways, including:

1. Adjacency Matrix
   The graph in the adjacency matrix representation can be written as A = [a_{ij}] with i being the self node and j being the target node. The matrix is written with the number 1 if i and j are adjacent, 0 if they are not adjacent, and 2 if it is a double edge.



	A	B	C	D
A	0	1	1	1
B	1	0	0	0
C	1	0	0	1
D	1	0	1	0

Fig 2.9 Example of adjacency matrix
(Source : Graph (Part 2) Slide by Dr. Rinaldi Munir)

2. Incidency Matrix
   The graph in the adjacency matrix representation can be written as A = [a_{ij}] with i being the self vertex and j being the target vertex. The matrix is written with the number 1 if i and j are adjacent and the direction is from i to j, 0 if they are not adjacent, and -1 if i and j are adjacent and the direction is i from j.

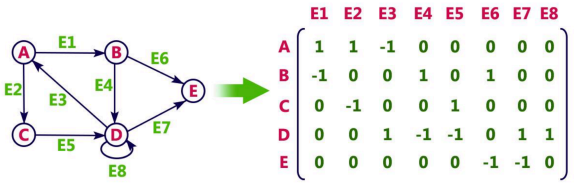


Fig 2.10 Example of incidency matrix
(Source : Graph (Part 2) Slide by Dr. Rinaldi Munir)

3. Adjacency List
   This representation directly describes which nodes are neighbors of other nodes in the form of a list.

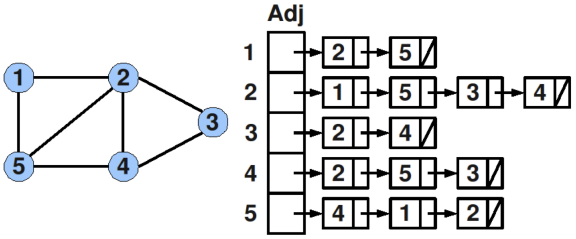


Fig 2.11 Example of adjacency list
(Source : Graph (Part 2) Slide by Dr. Rinaldi Munir)

C. Time Complexity

In algorithm complexity, there are two parameters to calculate the efficiency of a program, namely from space and from time. In this study, only one parameter will be used, namely time complexity. The main work of this time complexity is to calculate the stages of the algorithm. The number of stages of this algorithm is calculated from the number of operations performed as the input size (n).

Usually, time complexity is defined as T(n) which describes the maximum amount of time for a computation of size n. There is also a notation called Big-O which is written as O(f(n)), this notation is an asymptotic time complexity notation where f(n) is an upper bound of T(n) for larger n. There are groupings of algorithms from Big-O notation from good to bad, namely constant (O(1)), logarithmic (O(log n)), linear (O(n)), logarithmic linear (O(n log n)), quadratic (O(n^2)), cubic (O(n^3)), exponential (O(2^n)), and factorial (O(n!)).

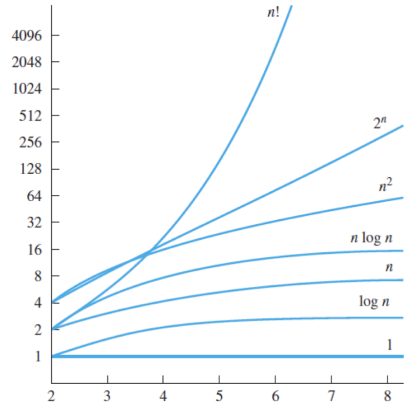


Fig 2.12 Time Complexity Graph
(Source : Algorithm Complexity (Part 2) Slide by Dr. Rinaldi Munir)

In this simple algorithm, there are several calculations in calculating the time complexity of this program.

1. PageRank
   The time complexity of PageRank is O(k . (|V| + |E|) where k is the number of iterations, V is the number of nodes (videos and users), and E is the number of edges (interactions) (Brin & Page, 1998).
2. Sorting and Interaction
   The time complexity of the sorting system is O(n log n) where n is the number of videos available (Cormen et al., 2009). Then, it will increase linearly from the preference with a value of O(n). The increase is also obtained from the existing interactions with a value of O(1).

III. Implementation Method

This section will explain how the For Your Page algorithm program is simulated in python with some information. This study uses some data as follows:

1. Video Data
   A video has its own data like on TikTok, namely video id, video title, video genre, viewers, likes, comments, shares, and saves.
2. User Interaction
   A video that passes For Your Page can be given interactions that will be counted in its score, the things that are assessed are whether the user likes, comments, shares, or saves and whether the user watches the video in full or in part or skips the existing video.

In the simple algorithm created, video interaction with users is symbolized by a directed graph (digraph) as well as a weighted graph. The directed graph here displays the direction between the user and the video in the dataset. For the weights on each side, the value of the FYP probability calculation score is depicted. In python, we can use the networkx library for using directed graphs and pandas for reading csv dataset.

```python
class TikTokFYP:
    def __init__(self, dataset_path):
        self.graph = nx.DiGraph()
        self.videos = []
        self.users = []
        self.genre_scores = {}
        self.video_data = pd.read_csv(dataset_path)
        self.calculate_initial()
```

Fig 3.1 Class and graph initialization

A. Data Processing

This study uses 30 videos stored as a dataset in a csv file using 5 types of genres to facilitate this research.

```csv
id,title,genre,viewers,likes,comments,shares,saves
1,Funny Cat Video,Animal,1000000,150000,10000,5000,20000
2,Valorant Clip,Game,800000,80000,5000,3000,10000
3,New Trend TikTok Dances,Entertainment,700000,50000,8000,2000,15000
4,Jedag Jedug,Music,650000,55000,7000,2500,12000
5,About Stocks,Education,400000,20000,4000,1000,8000
6,About Academic,Education,100000,5000,2000,500,3000
7,Viral Meme,Entertainment,300000,25000,5000,1500,7000
8,News,Entertainment,50000,3000,1000,200,800
9,Movie Clip,Entertainment,100000,7000,2000,600,1500
10,Lirik Lagu,Music,120000,10000,3000,700,2000
11,A Day in My Life,Entertainment,150000,12000,2500,800,4000
12,Beauty Hacks,Game,120000,10000,2000,700,3000
13,Travel Vlog,Entertainment,250000,20000,5000,1500,8000
14,Gadget Unboxing,Music,180000,15000,4000,1200,5000
15,Healthy Recipe,Education,350000,30000,6000,2000,10000
16,Funny Prank,Animal,180000,14000,3000,1000,4000
17,Gameplay PUBG,Game,150000,12000,2500,800,3000
18,Challenge Dance,Entertainment,400000,30000,7000,2500,10000
19,Remix Musik,Music,200000,18000,4000,1500,5000
20,Educational Webinar,Education,80000,4000,1500,400,2000
21,Animal Rescue,Animal,220000,20000,4000,1500,7000
22,Esports Championship,Game,900000,75000,10000,5000,20000
23,Flashmob Dance,Entertainment,320000,25000,5000,2000,9000
24,Rock Concert,Music,300000,28000,7000,2500,12000
25,Tech Talk,Education,150000,10000,3000,800,4000
26,Pet Show,Animal,400000,35000,8000,2500,12000
27,Game Review,Game,200000,18000,3000,1000,5000
28,Street Dance,Entertainment,300000,25000,6000,2000,10000
29,Indie Band Performance,Music,400000,35000,5000,1500,12000
30,Online Course,Education,120000,10000,2000,700,3000
```

Fig 3.2 Video dataset on csv

After the dataset is read, the dataset will be processed through a function called calculate_initial which will assess viewers, likes, comments, shares, and saves of a video. These components will be processed with their respective multipliers according to the assumed logical weight. For viewers, a multiplier of 0.125 will be used, likes with 0.5, comments with 0.25, shares with 0.5, and saves with 0.25. Then, if it has been initialized, a node is created in the graph implementation.

```python
def calculate_initial(self):
    for _, row in self.video_data.iterrows():
        video_id = row['id']
        genre = row['genre']
        if video_id not in self.graph:
            video_attributes = row.to_dict()
            video_attributes.pop('genre', None)
            initial_score = (
                row['viewers'] * 0.125 +
                row['likes'] * 0.5 +
                row['comments'] * 0.25 +
                row['shares'] * 0.5 +
                row['saves'] * 0.25
            )
            self.graph.add_node(video_id, type="video", genre=genre, initial_score=initial_score, **video_attributes)
            self.videos.append(video_id)

            # Add initial score to genre_scores
            if genre not in self.genre_scores:
                self.genre_scores[genre] = 0
            self.genre_scores[genre] += initial_score
```

Fig 3.3 Data calculation

B. User Interaction Calculation

A function needs to be created to calculate the score resulting from the interaction between the user and the video they are watching. The components in this interaction can be seen from likes, comments, shares, saves, and watch time. There is a multiplier for each component that adjusts to the overall weight when combined with data initialization processing. The weight for likes will be multiplied by 1.5, comments multiplied by 1, shares multiplied by 1.25, saves multiplied by 1.5, and watch time multiplied by 1. Keep in mind that this weighting greatly affects the user's FYP compared to the weighting according to the dataset. Weighting in dataset processing will only display videos based on the level of virality of the existing video.

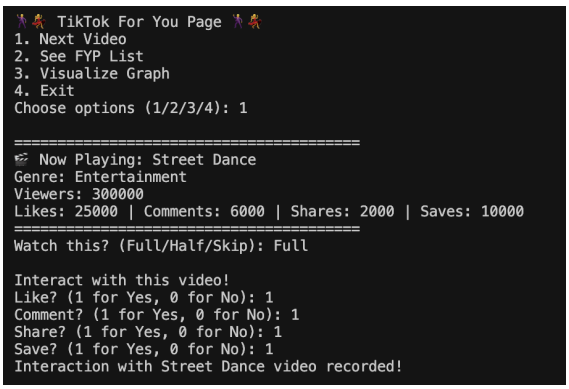
```python
def interact_with_video(self, user_id, video_id, like=0, comment=0, share=0, saves=0, full_watch=0):
    if video_id in self.videos:
        # Hitung bobot untuk interaksi
        weight = (like * 1.5 + comment * 1 + share * 1.25 + saves * 1.5 + full_watch * 1.0)
        genre = self.graph.nodes[video_id]['genre']

        # Update atau inisialisasi weight pada edge
        if self.graph.has_edge(user_id, video_id):
            if 'weight' in self.graph[user_id][video_id]:
                self.graph[user_id][video_id]['weight'] += weight
            else:
                self.graph[user_id][video_id]['weight'] = weight
        else:
            self.graph.add_edge(user_id, video_id, weight=weight)

        # Perbarui preferensi genre pengguna
        user_preferences = self.graph.nodes[user_id].setdefault('genre_preferences', {})
        user_preferences[genre] = user_preferences.get(genre, 0) + weight

        # Rehitung skor menggunakan PageRank
        self.recalculate_scores()
    else:
        print("Video not found.")
```

Fig 3.4 Interaction calculation



```
🕺💃 TikTok For You Page 🕺💃
1. Next Video
2. See FYP List
3. Visualize Graph
4. Exit
Choose options (1/2/3/4): 1

========================================
📽️ Now Playing: Street Dance
Genre: Entertainment
Viewers: 300000
Likes: 25000 | Comments: 6000 | Shares: 2000 | Saves: 10000
========================================
Watch this? (Full/Half/Skip): Full

Interact with this video!
Like? (1 for Yes, 0 for No): 1
Comment? (1 for Yes, 0 for No): 1
Share? (1 for Yes, 0 for No): 1
Save? (1 for Yes, 0 for No): 1
Interaction with Street Dance video recorded!
```

Fig 3.5 Interface for interaction

C. Scoring and PageRank

After the components for the score are complete, namely from the dataset and user interaction, a score needs to be created for each existing video based on the interaction. This score measurement feature uses the help of a function called PageRank. PageRank is a function to determine the level of importance in a graph. PageRank in this study uses two calculation components, as follows:

1. Edge weights
   This weight is obtained from the interaction between the user and the video they are watching based on the calculations mentioned earlier.
2. Personalization
   This is the initial distribution of the videos in the dataset based on the components mentioned earlier.

```python
def recalculate_scores(self):
    # Recalculate genre-based scores
    personalization = {
        video: self.graph.nodes[video]['initial_score']
        for video in self.videos
    }
    total_score = sum(personalization.values())

    # Normalize personalization for PageRank
    if total_score > 0:
        personalization = {k: v / total_score for k, v in personalization.items()}

    pagerank_scores = nx.pagerank(self.graph, weight='weight', personalization=personalization)

    for video_id, score in pagerank_scores.items():
        if video_id in self.graph.nodes and self.graph.nodes[video_id].get('type') == 'video':
            self.graph.nodes[video_id]['current_score'] = score
```

Fig 3.6 PageRank implementation on scoring

Then, a function is created that will combine PageRank with user preferences that will calculate the ranking of a video's possible FYP. This function also affects the interaction between users and the video genre. So, if a user interacts with a video, other videos with the same genre will also be affected by their score. For each existing FYP recommendation, the PageRank score will also be adjusted according to the user's preference factor in that genre. Then, the resulting FYP list will be taken as many as top_k. This score is also a factor that influences the randomization of videos through FYP, the greater the score, the more likely it is that a genre or video will appear in the user's FYP.

```python
def for_your_page(self, user_id, top_k=5):
    if user_id not in self.graph:
        return []

    # Recalculate scores to ensure they are up-to-date
    self.recalculate_scores()

    # Get user genre preferences
    user_preferences = self.graph.nodes[user_id].get('genre_preferences', {})

    # Generate recommendations
    recommendations = sorted(
        (
            (node, self.graph.nodes[node].get('current_score', 0) * (1 + user_preferences.get(self.graph.nodes[node].get('genre'), 0) / 10))
            for node in self.videos
        ),
        key=lambda x: x[1],
        reverse=True
    )
    # Add edge by score
    for video, score in recommendations:
        if not self.graph.has_edge(user_id, video):
            self.graph.add_edge(user_id, video, recommendation_score=score)

    return recommendations[:top_k]
```

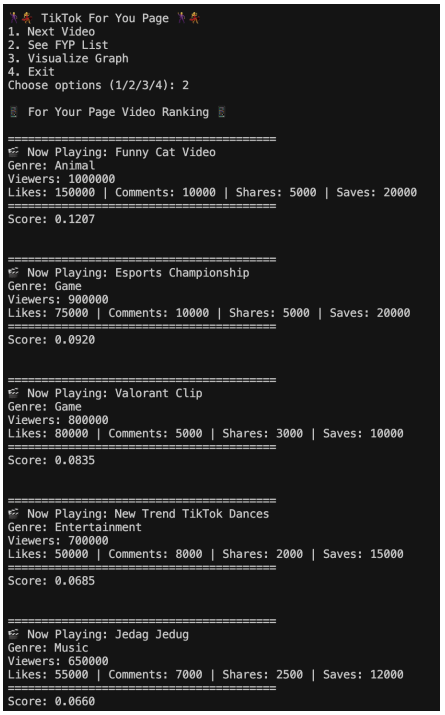
Fig 3.7 Function for FYP from PageRank and user preference

D. Display FYP and Graph Visualization

The score obtained from the previous function can be displayed by the user to see the video sequence with the highest possible FYP.

```python
def display_score(video_data):
    print("\n" + "="*40)
    print(f"📽️ Now Playing: {video_data['title']}")
    print(f"Genre: {video_data['genre']}")
    print(f"Viewers: {video_data['viewers']}")
    print(f"Likes: {video_data['likes']} | Comments: {video_data['comments']} | Shares: {video_data['shares']} | Saves: {video_data['saves']}")
    print("="*40)
```

Fig 3.8 Display FYP Implementation



```
🕺💃 TikTok For You Page 🕺💃
1. Next Video
2. See FYP List
3. Visualize Graph
4. Exit
Choose options (1/2/3/4): 2

🎥 For Your Page Video Ranking 🎥

========================================
📽️ Now Playing: Funny Cat Video
Genre: Animal
Viewers: 1000000
Likes: 150000 | Comments: 10000 | Shares: 5000 | Saves: 20000
========================================
Score: 0.1207


========================================
📽️ Now Playing: Esports Championship
Genre: Game
Viewers: 900000
Likes: 75000 | Comments: 10000 | Shares: 5000 | Saves: 20000
========================================
Score: 0.0920


========================================
📽️ Now Playing: Valorant Clip
Genre: Game
Viewers: 800000
Likes: 80000 | Comments: 5000 | Shares: 3000 | Saves: 10000
========================================
Score: 0.0835


========================================
📽️ Now Playing: New Trend TikTok Dances
Genre: Entertainment
Viewers: 700000
Likes: 50000 | Comments: 8000 | Shares: 2000 | Saves: 15000
========================================
Score: 0.0685


========================================
📽️ Now Playing: Jedag Jedug
Genre: Music
Viewers: 650000
Likes: 55000 | Comments: 7000 | Shares: 2500 | Saves: 12000
========================================
Score: 0.0660
```

Fig 3.9 Display FYP Interface

Then, the existing graph can be displayed with the help of the matplotlib library to produce a graph depiction with nodes in the form of user IDs and video IDs in the dataset.

```python
def visualize_graph(graph):
    pos = nx.spring_layout(graph)
    plt.figure(figsize=(12, 8))

    node_colors = []
    node_labels = {}
    for node, data in graph.nodes(data=True):
        if data['type'] == 'user':
            node_colors.append('blue')
            node_labels[node] = node  # ID User
        elif data['type'] == 'video':
            node_colors.append('green')
            node_labels[node] = node  # ID Video

    # Weight
    edge_weights = []
    edge_labels = {}
    for u, v, data in graph.edges(data=True):
        score = data.get('recommendation_score', data.get('weight', 0))  # Use Score
        edge_weights.append(score)
        edge_labels[(u, v)] = f"{score:.4f}"

    # GNode
    nx.draw_networkx_nodes(
        graph, pos,
        node_size=500,
        node_color=node_colors,
    )

    # Edge
    nx.draw_networkx_edges(
        graph, pos,
        width=[(w / max(edge_weights)) * 5 for w in edge_weights] if edge_weights else 1,
        alpha=0.6,
    )

    # Node label
    nx.draw_networkx_labels(graph, pos, labels=node_labels)

    # Edge label
    nx.draw_networkx_edge_labels(
        graph, pos,
        edge_labels=edge_labels,
        font_size=8,
    )

    plt.title("TikTok FYP Interaction Graph with Updated Scores")
    plt.axis("off")
    plt.show()
```

Fig 3.10 Graph visualization Implementation

IV. Implementation Testing and Result Discussion

A. Initial View of the Graph

If the user has just run the FYP program, the score for each video will be based on the calculation results from the dataset.

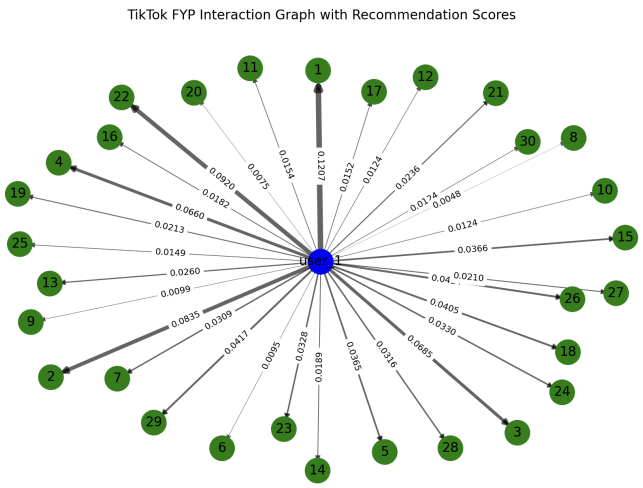


Fig 4.1 Graph visualization of initial data from visualize_graph function

B. User Interaction with Video

```
TikTok For You Page
1. Next Video
2. See FYP List
3. Visualize Graph
4. Exit
Choose options (1/2/3/4): 1

========================================
Now Playing: Esports Championship
Genre: Game
Viewers: 900000
Likes: 75000 | Comments: 10000 | Shares: 5000 | Saves: 20000
========================================
Watch this? (Full/Half/Skip): Full

Interact with this video!
Like? (1 for Yes, 0 for No): 1
Comment? (1 for Yes, 0 for No): 1
Share? (1 for Yes, 0 for No): 1
Save? (1 for Yes, 0 for No): 1
Interaction with Esports Championship video recorded!
```

Fig 4.2 User interaction with video

Users who have interacted with a video will increase the score of the video. In this case, the user interacts with a video titled "Esport Championship" with the genre "Game" and the score of the video will increase. Not only the score on the video, the score on videos with the same genre will also increase.

```
========================================
Now Playing: Esports Championship
Genre: Game
Viewers: 900000
Likes: 75000 | Comments: 10000 | Shares: 5000 | Saves: 20000
========================================
Score: 0.0920

========================================
Now Playing: Valorant Clip
Genre: Game
Viewers: 800000
Likes: 80000 | Comments: 5000 | Shares: 3000 | Saves: 10000
========================================
Score: 0.0835
```

```
========================================
Now Playing: Esports Championship
Genre: Game
Viewers: 900000
Likes: 75000 | Comments: 10000 | Shares: 5000 | Saves: 20000
========================================
Score: 0.1494

========================================
Now Playing: Valorant Clip
Genre: Game
Viewers: 800000
Likes: 80000 | Comments: 5000 | Shares: 3000 | Saves: 10000
========================================
Score: 0.1357
```

Fig 4.3 Score update, left is before the interaction and right is after the interaction

Users who do not interact with a video, by directly skipping the video they are watching, will not affect the score of the video or genre.

```
TikTok For You Page
1. Next Video
2. See FYP List
3. Visualize Graph
4. Exit
Choose options (1/2/3/4): 1

========================================
Now Playing: Valorant Clip
Genre: Game
Viewers: 800000
Likes: 80000 | Comments: 5000 | Shares: 3000 | Saves: 10000
========================================
Watch this? (Full/Half/Skip): Skip
Interaction with Valorant Clip video recorded!
```

Fig 4.4 User do not interact with the video

```
========================================
Now Playing: Valorant Clip
Genre: Game
Viewers: 800000
Likes: 80000 | Comments: 5000 | Shares: 3000 | Saves: 10000
========================================
Score: 0.0835
```

```
========================================
Now Playing: Valorant Clip
Genre: Game
Viewers: 800000
Likes: 80000 | Comments: 5000 | Shares: 3000 | Saves: 10000
========================================
Score: 0.0835
```

Fig 4.5 Score update if user do not interact with the video

C. For Your Page by Score

In this test, the researcher has interacted with 5 videos (like, comment, share, save, and full watch) with the genre "Animal" and produced the following graph display. Rank 1, 2, and 4 has the genre “Animal”.

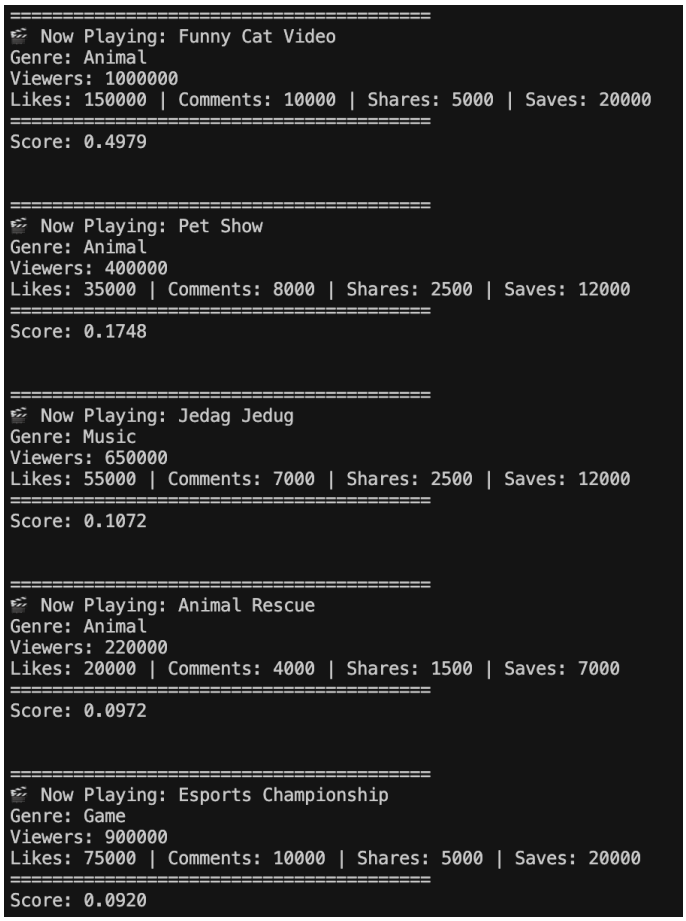


Fig 4.6 Top 5 with 3 "Animal" genre

This time a test will be carried out to see whether the FYP page in the 10 searches comes up with the most videos of the "Animal" genre. If yes, then the program is running in accordance with the objectives of the FYP itself.

Video	Genre
Pet Show	Animal
Flashmob Dance	Entertainment
Rock Concert	Music
Pet Show	Animal
Pet Show	Animal
Valorant Clip	Game
About Stocks	Education
Funny Animal Prank	Animal
Funny Animal Prank	Animal
Animal Rescue	Animal

Fig 4.7 10 FYP video

Here it can be seen, from 10 videos that passed through FYP, 6 videos were obtained that had the genre "Animal". It can be concluded that this algorithm is in accordance with the needs in the FYP algorithm simulation on TikTok.

In accordance with the calculation formula mentioned earlier, here is the time complexity calculation of the PageRank section.

$$O(k \cdot (|V| + |E|) = O(20 \cdot (31 + 4) = O(700)$$

This result will be added to the complexity calculation of the number of interactions with the existing video, which is 4.

$$O(700) + O(m) = O(700) + O(4) = O(704)$$

Then there are calculations in the sorting algorithm in this program.

$$O(n \log n) = O(30 \log_2 30) = O(147.3)$$

V. Conclusion

Modeling the For Your Page system on TikTok is very possible if using an algorithm using graph theory. In the graph, it will describe the interaction between users and the videos in the dataset. Thus, a video recommendation system is obtained that passes through FYP by referring to the interactions in the graph that has been created.

However, in time complexity, the program created actually has an expensive weight because there is PageRank used on a large graph. In the designed program it feels fast in execution time, but it is different if the existing dataset has a very large number because its time complexity will increase significantly on a large scale.

VI. Appendix

The complete sudoku solver program and other functions used can be found below.

https://github.com/fithrarzk/Makalah-Matdis

Below is a video demonstration and explanation of this project.

https://youtu.be/gD1TJAEu2tI

VII. Acknowledgment

All praise and gratitude are offered to the presence of the Almighty God, Allah Subhanahu wa Ta'ala, who has given the author the opportunity to complete the paper entitled "Modeling the Tiktok 'For Your Page' Algorithm Using Graph Theory and Algorithm Complexity". In addition, the author would like to express his deepest gratitude to the lecturer in charge of the Linear Algebra and Geometry course, Dr. Ir. Rinaldi Munir, M.T., for the lessons and motivation that have been given during the lecture.

References

[1] Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7), 107-117.

[2] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). MIT Press.

[3] R. Munir, "Graf Bagian 1" IF1220 Matematika Diskrit. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf [Accessed 4 January 2025]

[4] R. Munir, “Graf Bagian 2” IF1220 Matematika Diskrit. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/21-Graf-Bagian2-2024.pdf [Accessed 4 January 2025]

[5] R. Munir, “Graf Bagian 3” IF1220 Matematika Diskrit. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/22-Graf-Bagian3-2024.pdf [Accessed 4 January 2025]

[6] R. Munir, “Kompleksitas Algoritma Bagian 2” IF1220 Matematika Diskrit. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/26-Kompleksitas-Algoritma-Bagian2-2024.pdf [Accessed 4 January 2025]

Statement of Originality

I hereby declare that this paper is my own writing, not an adaptation, or translation of someone else's paper, and not plagiarized.

Bandung, 4 Januari 2024

Muhammad Fithra Rizki - 13523049