Optimalisasi Pemilihan Draft Hero Mobile Legends: Bang Bang dengan Graf Berbobot

Ahsan Malik Al Farisi - 13523074[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]themalique1910@gmail.com, 13523074@std.stei.itb.ac.id

***Abstract*— Industri *game online* di Indonesia kerap meningkat dengan salah satu *game mobile* yang populer di kalangan remaja adalah Mobile Legends: Bang Bang (MLBB) dari jenis Multiplayer Online Battle Arena (MOBA). Di dalam permainannya terdapat banyak unsur yang harus diperhatikan ketika menyusun strategi, salah satu unsur yang krusial adalah pemilihan hero mengingat banyak sekali atribut, skill, dan komposisi yang dapat dipilih dan dibentuk dari suatu tim. Makalah ini membahas akan bagaimana pemilihan hero dapat dioptimalkan dengan mengaplikasikan konsep graf berbobot. Dengan menggunakan data API dan mengekstrak data bobot dari Web MLBB dapat didefinisikan bobot relasi antar hero. Data ini kemudian diolah untuk pada akhirnya dapat dipilih hero dengan skor terbaik berdasarkan skor relasi *counter* dan kompatibilitasnya. Hasil penelitian ini dapat memberikan panduan kasar akan strategi dalam memilih hero di aras kompetitif.**

***Keywords*— Mobile Legends: Bang Bang, MLBB, Graf, Graf Berbobot, Draft Pick Hero, Web Scraping**

I. Pendahuluan

Pada masa ini game merupakan salah satu hal yang paling populer di kalangan remaja bahkan hingga dewasa. Game yang awalnya berasal dari mesin arcade dengan tombol dan joystick untuk bermain sekarang telah berkembang menjadi game online yang dapat dimainkan di device manapun. Game online terbagi menjadi beberapa jenis tergantung gawainya pertama ada game console seperti PS atau XBOX, game PC (Personal Computer), dan game mobile yaitu game handphone. Game mobile pun memiliki perkembangannya sendiri dan juga memiliki banyak jenis game, seperti RPG (Role Playing Games), FPS (First Person Shooter), MOBA (Multiplayer Online Battle Arena), dan masih banyak lagi.

Data pemain game di Indonesia berdasarkan dari Kementerian Komunikasi dan Informatika, pada tahun 2021 saja sudah menyentuh 121,7 juta sedangkan pada 2022 meningkat menjadi 174,1 juta. Angka ini kian meningkat dengan prediksi pada tahun 2025 pemain game akan berjumlah 191,2 juta orang. Salah satu contoh game yang sangat populer di kalangan remaja Indonesia adalah Mobile Legend : Bang Bang (MLBB) dengan jenis game MOBA dengan jumlah pemain aktif bulanan berjumlah 51 juta pemain berdasarkan data dari Moonton Indonesia.

Gambar 1. Logo Mobile Legend : Bang Bang
Sumber : wikipedia

Pemain game pada umumnya bermain untuk mendapatkan kesenangan, namun semakin lama pemain tersebut bermain kemampuannya pun akan meningkat hingga pada suatu titik permainan tersebut bukan lagi sekedar untuk mendapatkan kesenangan semata tetapi menjadi suatu hal yang kompetitif. Tentu saja dalam setiap permainan pasti memiliki strateginya sendiri, namun pada game MOBA yang mengandung banyak unsur dalam pemilihan strateginya seperti pemilihan hero, pemilihan *item*, pemilihan *emblem,* dan masih banyak lagi. Dengan demikian pemain harus dapat memilih dengan baik komposisi hero di awal permainan agar memiliki kemungkinan menang yang paling besar. Pemilihan hero di Mobile Legend pun memiliki banyak unsur seperti role, komposisi, dan counter. Sehingga keputusan untuk memilih suatu hero di tingkat yang kompetitif tidak lagi begitu mudah.

Dalam pemilihannya kita dapat mengimplementasikan dasar teori dari beberapa mata kuliah agar dapat menentukan pemilihan hero yang terbaik. Salah satu contoh implementasi yang dapat dilakukan adalah dengan melakukan perhitungan dengan graf berbobot agar kita dapat melihat bobot dan hubungan antar hero agar dapat dikalkulasikan hero mana yang memiliki skor terbaik untuk dapat dipilih.

II. Tujuan Makalah

Penelitian ini akan berfokus pada bagaimana pemain Mobile Legend : Bang Bang dapat mengoptimalisasi pemilihan hero dalam tim nya dengan pengaplikasian *weighted graph* agar mendapatkan komposisi yang terbaik dari sekian banyak hero yang tersedia di game nya.

III. Dasar Teori

A. Konsep Permainan Mobile Legends : Bang Bang

Mobile Legends : Bang Bang adalah game MOBA atau Multiplayer Online Battle Arena yang dikembangkan oleh Moonton dengan sistem permainan 5 vs 5 dan masing-masing tim memiliki tujuan untuk dapat menghancurkan *Base* utama

musuh. Pemenang dari permainan ini ditentukan oleh tim siapa yang paling pertama menghancurkan *Base* utama musuh.

Di dalam permainannya terdapat peta yang mengandung tiga jalur utama yaitu *Top Lane*, *Mid Lane*, dan *Bottom Lane*. Jalur utama ini akan dilalui oleh minion yang akan membantu kita dalam menyerang *turret* musuh dan pada tiap *lane* terdapat 3 *turret.* Selain dari jalur utama tersebut terdapat juga unsur lain seperti *jungle* yang terdiri atas *creep* yang dapat membantu tim kita untuk meningkatkan level dan mendapatkan emas. Juga terdapat entitas lain seperti *Turtle* yang akan memberikan emas dan *buff* untuk tim kita dan *Lord* yang dapat ikut membantu mendorong turret musuh.

Gambar 2. Peta Mobile Legend

Dari objektif dan peta terdapat juga komponen utama yaitu hero yang akan mengisi jalur-jalur dari peta ini dan akan memanfaatkan seluruh sumber daya yang ada di dalam peta untuk mendapatkan kemenangan.

B. Hero di Mobile Legend

Mobile Legends : Bang Bang saat ini memiliki jumlah hero sebanyak 127 (per 4 Januari 2025) dengan hero paling pertamanya adalah Miya dan hero yang terbarunya adalah Lukas. Dari 127 hero tersebut dapat kita kelompokkan lagi menjadi beberapa role yang terdapat di dalam permainannya yaitu Marksman, Tank, Fighter, Assassin, dan Mage. Tiap role memiliki keunikan, kelebihan, dan kekurangannya masing-masing yang dapat saling melengkapi suatu komposisi tim. Dari tiap hero pun sebenarnya memiliki skill seperti *damage*, *AOE* (Area of Effect), dan CC (Crowd Control) juga atribut yang beragam seperti kecepatan menembak, kelincahan, jumlah darah, jumlah *physical* atau *magical defense,* dan masih banyak lagi. Tiap skill dan atribut juga dapat menjadi pelengkap komposisi tim ataupun dapat dipakai untuk mengincar hero lawan.

Gambar 3. Hero di Mobile Legend
Sumber : Web VCGamers

Pemilihan dari hero juga ada yang namanya *counter* atau melawan hero musuh secara spesifik. Misalnya musuh memilih hero *fighter* yang mengincar musuhnya dengan jarak dekat, sehingga hero yang memiliki kelincahan yang rendah akan susah kabur melawan hero *fighter.* Namun, kita harus tetap mengisi *role mage,* sehingga setelah melihat musuh memilih *fighter* jarak dekat kita dapat memilih *hero counter* Novaria yang memiliki jangkauan tembak yang tinggi dan juga kelincahan yang tinggi sehingga *fighter* dari musuh tidak dapat menjangkau kita dengan mudah. Namun dengan kita memilih Novaria kita juga dapat di-*counter* oleh *assassin* yang lincah dan memiliki *damage burst* yang kuat karena pada dasarnya mage memiliki daya tahan yang rendah. Hal tersebut merupakan salah satu skenario dari mengapa pemilihan hero merupakan hal yang penting di dalam game, karena kita dapat memilih hero yang sesuai agar dapat memanfaatkan kelemahan yang dimiliki oleh musuh.

C. Draft Pick Hero

Dalam Mobile Legend terdapat beberapa pilihan mode game dan yang paling umum adalah *classic* dan *ranked.* Perbedaan utama dari keduanya adalah pada mode *classic* tidak terdapat *ban* hero atau sistem agar hero tertentu tidak dapat dipilih baik oleh tim kita maupun oleh tim musuh. Sedangkan pada mode *ranked* terdapat sistem *ban* hero sesuai dengan tingkatan *rank,* misalnya pada *rank Epic* tiap tim dapat *ban* tiga hero, pada *rank Legend* tiap tim dapat *ban* empat hero, dan terakhir pada *rank Mythic* tiap tim dapat *ban* lima hero. Pada makalah ini akan dibahas pada sistem permainan ranked di *rank Mythic.* Adapun sistem *draft pick* hero seperti berikut dimisalkan pemilihan antara tim A sebagai tim pertama dan tim B sebagai tim kedua.

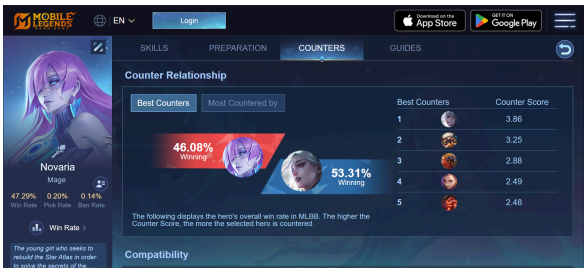
1. Fase Ban Hero
    a. Dua orang pada tim A akan melakukan ban hero.
    b. Dua orang pada tim B akan melakukan ban hero.
    c. Poin a dan b dilakukan sekali lagi sesuai urutan
    d. Akan tersisa satu pemain dari tiap tim untuk melakukan *ban* hero dimulai dari tim A dan terakhir oleh tim B.
    e. Hero yang sudah di-*ban* tidak dapat di-*ban* lagi dan tidak dapat dipilih saat fase pemilihan hero.
2. Fase Pemilihan Hero
    a. Satu orang pada tim A akan melakukan pilihan hero pertama (*first pick*).
    b. Dua orang pada tim B akan memilih dua hero.
    c. Selanjutnya akan dipilih dua hero mulai dari tim A dan bergantian dengan tim B hingga tersisa satu pemain lagi pada tim B.
    d. Pemain terakhir dari tim B akan memilih satu hero terakhir (*last pick).*
    e. Hero yang sudah dipilih tidak dapat dipilih lagi, sehingga memastikan komposisi tim yang antar tim nya unik.

Dalam tiap pergantian pemilihannya, skor kebagusan dari hero yang dapat dipilih akan berfluktuasi tergantung hero yang dipilih oleh teman kita juga oleh musuh kita. Sehingga sangat

penting untuk memperhatikan tiap pilihan hero.

D. Data Hero Mobile Legend

Mobile Legends : Bang Bang memiliki web resmi yang memuat berbagai informasi seperti *event* yang sedang berjalan, *patch notes* dari permainan, *esport* yang sedang berjalan, dan juga data-data hero yang mengandung statistik kemenangan dan juga relasi dari satu hero dengan satu hero lain. Misalnya relasi yang menyatakan hero A dapat di-*counter* dengan hero B atau hero A dapat bekerja baik dengan hero B di dalam satu tim. Data statistik ini yang memiliki skor dari setiap relasinya akan dijadikan dasar dari bobot pada implementasi *weighted graph.*

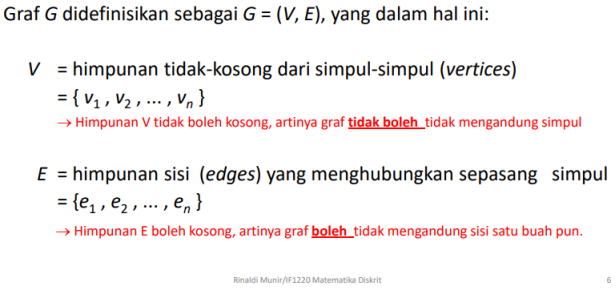


Gambar 4. Statistik Relasi Novaria di Web Mobile Legends
Sumber : Web MLBB

Data yang terkandung dari website tersebut sangat banyak dan tidak mungkin untuk dimasukkan secara manual dengan demikian akan dilakukan ekstraksi data dari web tersebut agar datanya dapat diolah dengan *weighted graph.* Detail dari ekstraksi data akan dijelaskan lebih lanjut pada bagian implementasi.

E. Graf Berbobot

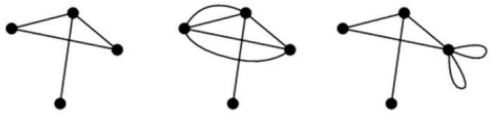
Graf kumpulan dari titik "*node*" atau "*vertex*" yang dapat terhubung dengan titik lain dengan garis "edge" sebagai penghubungnya. Graf biasanya digunakan untuk merepresentasikan objek diskrit dan hubungannya antar objek [3]. Secara matematis graf dapat didefinisikan sebagai berikut.



Gambar 5. Definisi Graf
Sumber : Graf (1) - Rinaldi Munir

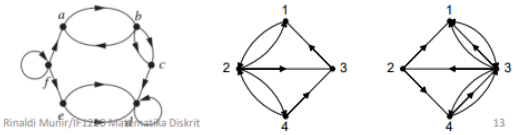
Suatu graf jika dilihat dari orientasi arah pada sisinya, graf dapat dibedakan atas 2 jenis, yaitu :

1. Graf tak berarah (*undirected graph*)
   Undirected graph adalah graf yang tidak mempunyai orientasi arah.



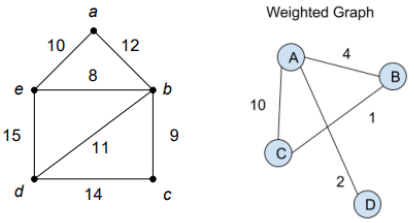
Gambar 6. Graf tak berarah
Sumber : Graf (1) - Rinaldi Munir

2. Graf berarah (*directed graph*)
   Directed graph adalah graf yang memiliki orientasi arah.



Gambar 7. Graf berarah
Sumber : Graf (1) - Rinaldi Munir

Selain dari itu terdapat jenis lain dari graf yaitu graf berbobot. Graf berbobot adalah graf yang setiap sisinya diberi sebuah harga.



Gambar 8. Graf berbobot
Sumber : Graf (1) - Rinaldi Munir

Dalam penerapannya untuk kita dapat menggabungkan beberapa jenis dari graf sesuai dengan kebutuhan dan kondisi yang ada. Misalnya menggabungkan graf berarah dengan graf berbobot sehingga terdapat hubungan yang jelas antara satu hero dengan hero yang lain juga agar dapat dikalkulasikan skor skor terbaiknya

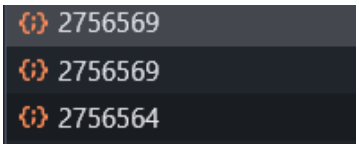
.

IV. Implementasi

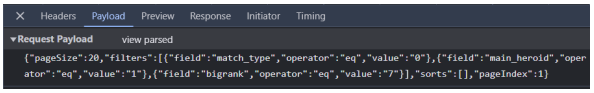
A. Ekstraksi Data

Pada makalah ini ekstraksi data dibagi menjadi tiga tahap, pertama yaitu mengambil data dari API (Application Programming Interface) yang tersedia di web dan mengandung id *main hero,* nama dan atribut *main hero,* urutan id *subhero* yang memiliki relasi *counter* (siapa yang bisa di-*counter* dan siapa yang meng-*counter*) dan relasi *compatibility* (siapa yang paling cocok dalam satu tim dan siapa yang paling tidak cocok) dengan *main hero*. Tahap kedua adalah ekstraksi data dengan mengambil bobot atau skor relasi yang terdapat pada web karena pada data pada bagian API tidak terdapat skornya hanya urutan *subhero* id yang memiliki skor tersebut. Tahap terakhir adalah mengekstraksi data yang diperlukan dari data yang sudah diambil dan memprosesnya agar dapat diolah dalam perhitungan graf.

Proses ekstraksi pertama dengan API dilakukan pertama dengan melakukan inspeksi pada web dan melihat *network tab* ketika kita membuka halaman "COUNTERS". Pada bagian XHR/Fetch akan terdapat 3 API yang diambil, dan pada tiap API terdapat *payload* nya yang harus kita kirim agar mendapatkan API yang diinginkan. Kita dapat menyesuaikan *payload* agar mendapatkan hero yang kita pilih dengan menyesuaikan hero id pada bagian "field", adapun list API dan payloadnya seperti berikut.



Gambar 9. List API pada *Page Counter*
Sumber : Network Tabs Web MLBB



Gambar 10. Payload API
Sumber : Network Tabs Web MLBB

Dengan demikian kita dapat mengekstraksi data yang terdapat pada 3 API tersebut dengan menyesuaikan url dan hero id yang akan dipilih. Untuk proses ekstraksi dilakukan dengan cuplikan kode berikut.

```python
match_types = [
    {"value": 0, "filename": "counter.json"},       # Counter relationships
    {"value": 1, "filename": "compatibility.json"}, # Compatibility relationships
]

for hero_id in range(1, 128):
    hero_dir = os.path.join(base_dir, f"hero_{hero_id}")
    os.makedirs(hero_dir, exist_ok=True)
    hero_payload={"pageSize":20,
                  "filters":[
                      {"field":"hero_id","operator":"eq","value":str(hero_id)}
                      ],
                  "sorts":[],
                  "pageIndex":1,
                  "object":[]}

    for match_type in match_types:
        score_payload = {
            "pageSize": 20,
            "filters": [
                {"field": "match_type", "operator": "eq", "value": str(match_type["value"])},
                {"field": "main_heroid", "operator": "eq", "value": str(hero_id)},
                {"field": "bigrank", "operator": "eq", "value": "7"}
            ],
            "sorts": [],
            "pageIndex": 1
        }

        try:
            response = requests.post(base_url+score_url, json=score_payload, headers=headers)
            response.raise_for_status()

            data = response.json()

            file_path = os.path.join(hero_dir, match_type["filename"])
            with open(file_path, "w") as json_file:
                json.dump(data, json_file, indent=4)

        except requests.exceptions.RequestException as e:
            print(f"Error fetching data for Hero ID {hero_id}, match type {match_type['value']}: {e}")


    try:
        response = requests.post(base_url+hero_url, json=hero_payload, headers=headers)
        response.raise_for_status()

        # Parse the JSON response
        data = response.json()

        # Save the data into a JSON file
        file_path = os.path.join(hero_dir, "hero.json")
        with open(file_path, "w") as json_file:
            json.dump(data, json_file, indent=4)

        print(f"Saved data for Hero ID {hero_id}")

    except requests.exceptions.RequestException as e:
        print(f"Error fetching data for Hero ID {hero_id}, match type {match_type['value']}: {e}")
```

Gambar 11. Implementasi Kode Ekstraksi API
Sumber : Dokumentasi Pribadi

Tahap ekstraksi data kedua adalah dengan menggunakan dua library yang dijalankan di python. Pertama adalah library BeautifulSoup yang berfungsi untuk mengambil data sesuai dengan struktur HTML yang dipilih, misalnya kita dapat mengambil data yang terkandung pada *div* dengan *class name* tertentu. Kedua adalah library Selenium untuk dapat menjalankan proses ekstraksi secara otomatis dengan cara membuka web dan melakukan proses pembukaan page secara otomatis.

Proses mengekstrak data dari web official Mobile Legends : Bang Bang memerlukan beberapa tahap yang cukup rumit dikarenakan struktur web yang di-*load* secara dinamis oleh JavaScript dan data skor yang terdapat tidak berada di *main page* sehingga strukturnya HTML-nya tidak dapat diekstrak secara langsung oleh BeautifulSoup. Sehingga proses ekstrak oleh Selenium harus menjalankan beberapa langkah agar dapat masuk ke *page counter* yang mengandung bobot dari hero untuk kemudian diekstrak. Adapun cuplikan dari implementasi kode yang dipakai untuk mengekstraksi ketika sudah terhubung dengan web nya menggunakan Selenium dan proses ekstraksi oleh BeautifulSoup.

```python
'''Change Component To Counter Page'''
    # Menunggu Counter tab dapat diklik
    counters_tab = WebDriverWait(driver, 20).until(
        EC.element_to_be_clickable((By.XPATH, "//div[contains(@class, 'mt-list-item') and span[text()='COUNTERS']]"))
    )
    counters_tab.click() # Klik page counter

    # Menunggu data muncul
    WebDriverWait(driver, 20).until(
        EC.presence_of_element_located((By.CLASS_NAME, 'mt-2684369'))  # Counter score class name
    )
    WebDriverWait(driver, 20).until(
        EC.presence_of_element_located((By.CLASS_NAME, 'mt-2684562'))  # Compatibility score class name
    )

    '''Scraping the data on the first component page'''
    html_content = driver.page_source
    soup = BeautifulSoup(html_content, "html.parser")

    first_counter_score = soup.find_all('div', class_='mt-2684369')
    first_compatibility_score = soup.find_all('div', class_='mt-2684562')
    print("First Component Score")
    for score in first_counter_score:
        print(score.text)
    for score in first_compatibility_score:
        print(score.text)
```

Gambar 12. Implementasi Kode Ekstraksi Struktur Web
Sumber : Dokumentasi Pribadi

```python
def main():
    base_url = "https://m.mobilelegends.com/hero/detail?channelid=2819992&heroid="
    base_dir = "hero_data"
    os.makedirs(base_dir, exist_ok=True)

    for hero_id in range(1, 128):  # 127 total hero
        url = base_url + str(hero_id)
        print(url)
        print(f"Scraping data for Hero ID: {hero_id}")
        hero_dir = os.path.join(base_dir, f"hero_{hero_id}")
        os.makedirs(hero_dir, exist_ok=True)

        try:
            first_counter_score, second_counter_score, first_compatibility_score, second_compatibility_score = scrape_dynamic_content(url)

            data = {
                "counter": [score.text for score in first_counter_score],
                "countered": [score.text for score in second_counter_score],
                "compatible": [score.text for score in first_compatibility_score],
                "incompatible": [score.text for score in second_compatibility_score],
            }

            file_path = os.path.join(hero_dir, "weight.json")
            with open(file_path, "w") as json_file:
                json.dump(data, json_file, indent=4)
        except Exception as e:
            print(f"Error scraping Hero ID {hero_id}: {e}")
```

Gambar 13. Implementasi Kode Main Ekstraksi Struktur Web
Sumber : Dokumentasi Pribadi

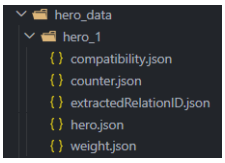
Data yang diambil dari struktur web tersebut akan menghasilkan struktur file JSON yang terdiri dari 4 jenis array yaitu *counter*, *countered*, *compatible*, dan *incompatible* dengan tiap array mengandung skor/bobot terurut dari *subhero*.

Tahap terakhir dari proses ekstraksi data adalah mengolah data mentah yang diambil dari API dan juga struktur web untuk kemudian digabungkan agar mendapatkan data yang dapat digunakan dalam perhitungan. Pertama kita harus mengekstrak data urutan *subhero* id pada relasi *counter* dan relasi *compatibility* yang terdapat di file JSON untuk kemudian dihasilkan 4 jenis array sama dengan yang dihasilkan dari ekstraksi struktur web. Namun pada data ini tiap array mengandung urutan *subhero* id. Adapun cuplikan implementasi untuk mengekstrak data dari API sebagai berikut.

```python
def extractIdScore(folderPath):
    counterId = []
    counteredId = []
    compatibleId = []
    incompatibleId = []

    counter_filepath = os.path.join(folderPath, "counter.json")
    compatibility_filepath = os.path.join(folderPath, "compatibility.json")
    with open(counter_filepath, "r") as f1, open(compatibility_filepath, "r") as f2:
        counters_data = json.load(f1)
        compatibility_data = json.load(f2)

    # Example: Iterating through sub_heroes
    counter_sub_heroes = counters_data['data']['records'][0]['data']['sub_hero']
    counter_sub_heroes_last = counters_data['data']['records'][0]['data']['sub_hero_last']
    for idx, hero in enumerate(counter_sub_heroes, start=1):
        counterId.append(hero['heroid'])
    for idx, hero in enumerate(counter_sub_heroes_last, start=1):
        counteredId.append(hero['heroid'])

    compatibility_sub_heroes = compatibility_data['data']['records'][0]['data']['sub_hero']
    compatibility_sub_heroes_last = compatibility_data['data']['records'][0]['data']['sub_hero_last']
    for idx, hero in enumerate(compatibility_sub_heroes, start=1):
        compatibleId.append(hero['heroid'])
    for idx, hero in enumerate(compatibility_sub_heroes_last, start=1):
        incompatibleId.append(hero['heroid'])

    return counterId, counteredId, compatibleId, incompatibleId
```

Gambar 13. Implementasi Kode Proses Data Mentah API
Sumber : Dokumentasi Pribadi

Kemudian setelah mengolah data kita dapat menggabungkan seluruh data yang tersedia pada tiap folder hero dengan compatibility.json, counter.json, dan hero.json mengandung data mentah hasil ekstraksi API, extractedRelationID.json mengandung urutan *subhero* Id hasil ekstraksi dari compatibility.json dan terakhir weight.json mengandung bobot/skor sesuai dengan urutan *subhero* id pada extractedRelationID.json. Seluruh file tersebut kemudian akan digabungkan menjadi satu bernama HeroData.json, adapun cuplikan struktur folder hero dan juga struktur data final yang terdapat di root folder.



Gambar 14. Struktur folder per hero
Sumber : Dokumentasi Pribadi

```json
[
    {
        "heroId": 1,
        "heroName": "Miya",
        "relationData": {
            "counter": [
                {
                    "hero_id": "3",
                    "value": "3.92"
                },
```

Gambar 15. Struktur HeroData.json
Sumber : Dokumentasi Pribadi

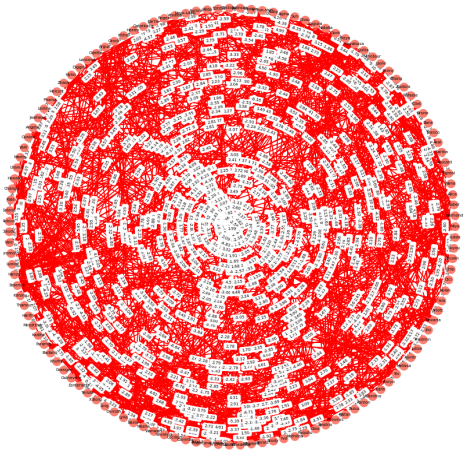
B. Pemodelan Graf

Dari data yang sudah diambil, tiap hero akan memiliki relasi dengan 20 hero lain. Kategori tersebut akan dibagi menjadi 2 graf, yaitu graf *counter* dan graf *compatibility*. Pada graf *counter* akan terdapat 5 relasi hero yang dapat di-*counter main hero* dan 5 relasi hero yang akan meng-*counter main hero.* Sedangkan pada graf *compatibility* akan terdapat 5 relasi hero yang bersinergi baik dengan *main hero* dan 5 relasi hero yang bersinergi buruk dengan *main hero.* Graf tersebut dibentuk dengan library networkx. Adapun cuplikan kode dari implementasinya seperti berikut.

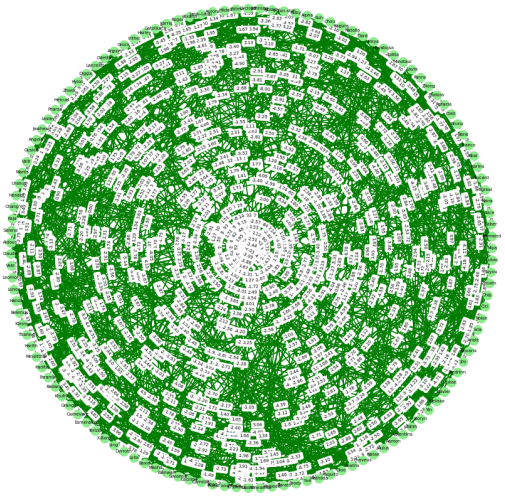
```python
'''INITIALIZE GRAPH'''
counterGraph = nx.DiGraph()
compatibilityGraph = nx.DiGraph()

heroes = []
counterEdges = []
compatibilityEdges = []

with open("HeroData.json", "r") as f1, open("heroDictionary.json", "r") as f2:
    hero_data = json.load(f1)
    hero_dictionary = json.load(f2)

for idx in range(127):
    heroName = hero_data[idx]['heroName']
    baseRelation = hero_data[idx]['relationData']
    heroes.append(heroName)

    for i in range(5):
        counterId = baseRelation['counter'][i]['hero_id']
        counterWeight = baseRelation['counter'][i]['value']
        subheroName = hero_dictionary[counterId]
        counterEdges.append((heroName, subheroName, counterWeight))

        counteredId = baseRelation['countered'][i]['hero_id']
        counteredWeight = baseRelation['countered'][i]['value']
        subheroNameCountered = hero_dictionary[counteredId]
        counterEdges.append((heroName, subheroNameCountered, counteredWeight))

        compatibleId = baseRelation['compatible'][i]['hero_id']
        compatibleWeight = baseRelation['compatible'][i]['value']
        subheroNameCompatible = hero_dictionary[compatibleId]
        compatibilityEdges.append((heroName, subheroNameCompatible, compatibleWeight))

        incompatibleId = baseRelation['incompatible'][i]['hero_id']
        incompatibleWeight = baseRelation['incompatible'][i]['value']
        subheroNameIncompatible = hero_dictionary[incompatibleId]
        compatibilityEdges.append((heroName, subheroNameIncompatible, incompatibleWeight))

counterGraph.add_nodes_from(heroes)
counterGraph.add_weighted_edges_from(counterEdges)

compatibilityGraph.add_nodes_from(heroes)
compatibilityGraph.add_weighted_edges_from(compatibilityEdges)
```

Gambar 16. Implementasi Struktur Graf
Sumber : Dokumentasi Pribadi

Setelah dimasukkan semua data dari simpul dan busur ke graf *counter* dan graf *compatibility.* Dapat dijalankan kode sederhana untuk memvisualisasikan kedua graf yang terbentuk.



Gambar 17. Visualisasi *Counter Graph*
Sumber : Dokumentasi Pribadi



Gambar 18. Visualisasi *Compatibility Graph*

Sumber : Dokumentasi Pribadi

C. Algoritma Pemilihan Hero

Setelah kita memiliki dua graf, yaitu graf *counter* dan graf *compatibility* kita dapat menggunakan kedua graf ini untuk mencari skor gabungan terbaik dari kedua graf. Pertama algoritma ini mengambil argumen hero dari kedua tim, kemudian tiap *hero* dari available_heroes (hero lain yang tidak terdapat di team_heroes dan opposing_team_heroes) akan dihitung skor counternya berdasarkan hubungan hero yang sedang dihitung dengan hero yang terdapat pada tim musuh. Kemudian akan dikalkulasikan juga compatibility_score dengan mencari total nilai sinergi yang dimiliki hero yang sedang dihitung dengan hero yang terdapat pada tim sendiri. Setelah itu skor *counter dan compatibility* akan digabungkan dan diurutkan, dengan demikian akan didapatkan hero terbaik yaitu yang memiliki sinergi terbaik dengan seluruh hero pada tim sendiri dan memiliki nilai counter terbaik dengan seluruh hero pada tim musuh.

```python
def calculate_scores(team_heroes, opposing_team_heroes, available_heroes):
    hero_scores = []
    for hero in available_heroes:
        counter_score = 0
        synergy_score = 0

        for counter in opposing_team_heroes:
            edge_data = counterGraph[hero].get(counter, {})
            weight = float(edge_data.get('weight', 0))
            counter_score += weight

        for compatible in team_heroes:
            edge_data = compatibilityGraph[hero].get(compatible, {})
            weight = float(edge_data.get('weight', 0))
            synergy_score += weight

        total_score = counter_score + synergy_score
        hero_scores.append((hero, total_score, counter_score, synergy_score))

    return hero_scores

def get_best_and_worst_picks(team_heroes, opposing_team_heroes, available_heroes):
    hero_scores = calculate_scores(team_heroes, opposing_team_heroes, available_heroes)
    sorted_heroes = sorted(hero_scores, key=lambda x: x[1], reverse=True)

    best_picks = sorted_heroes[:5]
    worst_picks = sorted_heroes[-5:]

    return best_picks, worst_picks
```

Gambar 19. Implementasi Algoritma Pemilihan Hero Terbaik dan Terburuk
Sumber : Dokumentasi Pribadi

Misal hero yang tersedia adalah dari A hingga Z, dengan tim kita sudah memilih hero A, sedangkan musuh sudah memilih hero Y dan Z. Pada algoritma tersebut akan dikalkulasikan nilai available_hero dari B hingga X. Ambil pada iterasi pertama yaitu pada hero B, di loop pertama hero B akan dihitung nilai relasi *counter*-nya dengan hero Y dan Z pada tim musuh dan ditotalkan. Kemudian pada loop kedua, hero dari B akan dihitung sinergi nya dengan hero A dan nilai sinergi akan disumasi dengan nilai counter untuk kemudian di-*append* ke dalam hero_scores. Proses ini berulang mulai dari hero B hingga X. Setelah semua dibandingkan maka akan diurutkan dan diberikan 5 terbaik dan terburuknya.

D. Simulasi Draft

Pada simulasi draft pertama akan dilakukan proses *ban* 10 hero dengan 5 ban dari masing-masing tim, proses ini akan menghapus nama hero dari available_heroes. Namun jumlah dari hero yang ingin di-*ban* dapat disesuaikan di dalam kode. Kemudian akan diadakan draft pick dengan total 6 sesi, sesi pertama yaitu *first pick* pada tim A, sesi kedua hingga kelima tiap tim akan memilih 2 hero, dan terakhir adalah *last pick* yaitu tim B akan memilih 1 hero terakhir. Pada tiap input akan dihitung counter_score dan compatibility_score nya dengan hero yang telah dipilih baik oleh tim sendiri maupun tim musuh. Sehingga selama 10 kali pemilihan hero akan ada 10 kali perhitungan untuk memberikan 5 pilihan terbaik dan 5 pilihan terburuk berdasarkan skor perhitungan. Adapun cuplikan implementasi dari simulasi draft seperti berikut.

```python
'''PICK PHASE'''
for team_name, picks_in_this_turn in pick_order:
    print(f"\n---------- PICK PHASE - {team_name} ----------")
    remaining_picks = picks_in_this_turn
    print(f"{team_name} is picking {remaining_picks} hero(s).")

    for _ in range(picks_in_this_turn):
        team_heroes = teams[team_name]
        opposing_team_name = "Team B" if team_name == "Team A" else "Team A"
        opposing_team_heroes = teams[opposing_team_name]

        best_picks, worst_picks = get_best_and_worst_picks(team_heroes, opposing_team_heroes, available_heroes)

        print("\nTop 5 Best Picks:")
        for hero, score, counter_score, synergy_score in best_picks:
            print(f"{hero} (Total: {score:.2f}, Counter: {counter_score:.2f}, Synergy: {synergy_score:.2f})")

        print("\nTop 5 Worst Picks:")
        for hero, score, counter_score, synergy_score in reversed(worst_picks):
            print(f"{hero} (Total: {score:.2f}, Counter: {counter_score:.2f}, Synergy: {synergy_score:.2f})")

        while True:
            print("\nPlease choose your pick from the best picks:")
            chosen_hero = input(f"Pick one hero: ")
            if chosen_hero in available_heroes:
                teams[team_name].append(chosen_hero)
                available_heroes.remove(chosen_hero)
                print(f"{team_name} picks {chosen_hero}")
                break
            else:
                print("Invalid pick.")
```

Gambar 20. Implementasi Drafting Hero
Sumber : Dokumentasi Pribadi

V. Eksperimen dan Pembahasan

Setelah kita mendapatkan simulasi drafting dan juga algoritma kalkulasi skor, kita akan menjalani programnya. Pada pilihan pertama karena belum ada data yang dapat dibandingkan maka tampilan *best pick* dan *worst pick* adalah sesuai urutan hero. Setelah itu misal pada tim A pertama kita akan memilih Miya maka akan dihasilkan 5 hero terbaik yang dapat dipilih oleh tim B untuk dapat meng-*counter* Miya dan juga 5 hero terburuk yang akan di-*counter* oleh Miya.

```
Please choose your pick from the best picks:
Pick one hero: Miya
Team A picks Miya

---------- PICK PHASE - Team B ----------
Team B is picking 2 hero(s).

Top 5 Best Picks:
Belerick (Total: 3.70, Counter: 3.70, Synergy: 0.00)
Granger (Total: 3.21, Counter: 3.21, Synergy: 0.00)
Thamuz (Total: 2.72, Counter: 2.72, Synergy: 0.00)
Lolita (Total: 2.68, Counter: 2.68, Synergy: 0.00)
Baxia (Total: 2.65, Counter: 2.65, Synergy: 0.00)

Top 5 Worst Picks:
Masha (Total: -2.39, Counter: -2.39, Synergy: 0.00)
Zilong (Total: -2.45, Counter: -2.45, Synergy: 0.00)
Irithel (Total: -2.49, Counter: -2.49, Synergy: 0.00)
Guinevere (Total: -2.80, Counter: -2.80, Synergy: 0.00)
Saber (Total: -3.92, Counter: -3.92, Synergy: 0.00)
```

Gambar 21. Eksperimen Pemilihan Hero ke-1
Sumber : Dokumentasi Pribadi

Kalau kita lihat dengan data yang tersedia pada web nya akan sesuai dengan hasil algoritma kita, yaitu *best pick* tim B akan terdiri dari hero yang dapat meng-*counter* Miya dengan baik, yaitu Belerick, Granger, Thamuz, Lolita dan Baxia. Sedangkan pada *worst pick* akan terdiri dari hero yang dapat Miya counter, yaitu Saber, Guinevere, Irithel, Zilong, dan Masha. Adapun cuplikan statistik data berikut yang diambil

dari web untuk melihat kesesuaiannya.

Gambar 22. Data Hero yang Miya *Counter*
Sumber : Web MLBB

Gambar 23. Data Hero yang meng-*counter* Miya
Sumber : Web MLBB

Jika kita lanjutkan programnya maka algoritma akan berjalan sesuai dengan yang telah dibahas pada bagian implementasi. Selanjutnya pada tim B kita memilih Belerick yang kemudian akan dihitung ulang skor terbaiknya untuk pemilihan hero kedua dari tim B. Setelah pemilihan kedua dari tim B yaitu Hanzo, maka saat tim A memilih akan dikalkulasikan ulang lagi hero mana yang terbaik berdasarkan hero yang sudah dipilih dari tim sendiri dan tim musuh.

```
Please choose your pick from the best picks:
Pick one hero: Belerick
Team B picks Belerick

Top 5 Best Picks:
Hanzo (Total: 4.39, Counter: 0.00, Synergy: 4.39)
Natalia (Total: 3.63, Counter: 0.00, Synergy: 3.63)
Angela (Total: 3.51, Counter: 0.00, Synergy: 3.51)
Granger (Total: 3.21, Counter: 3.21, Synergy: 0.00)
Thamuz (Total: 2.72, Counter: 2.72, Synergy: 0.00)

Top 5 Worst Picks:
Carmilla (Total: -2.61, Counter: 0.00, Synergy: -2.61)
Guinevere (Total: -2.80, Counter: -2.80, Synergy: 0.00)
Tigreal (Total: -2.97, Counter: 0.00, Synergy: -2.97)
Atlas (Total: -3.00, Counter: 0.00, Synergy: -3.00)
Saber (Total: -3.92, Counter: -3.92, Synergy: 0.00)
```

Gambar 24. Eksperimen Pemilihan Hero ke-2
Sumber : Dokumentasi Pribadi

```
Please choose your pick from the best picks:
Pick one hero: Hanzo
Team B picks Hanzo

---------- PICK PHASE - Team A ----------
Team A is picking 2 hero(s).

Top 5 Best Picks:
Natalia (Total: 8.37, Counter: 8.37, Synergy: 0.00)
Helcurt (Total: 7.15, Counter: 7.15, Synergy: 0.00)
Fanny (Total: 7.09, Counter: 5.24, Synergy: 1.85)
Gusion (Total: 6.58, Counter: 4.80, Synergy: 1.78)
Zilong (Total: 6.23, Counter: 6.23, Synergy: 0.00)

Top 5 Worst Picks:
Bane (Total: -6.14, Counter: -6.14, Synergy: 0.00)
Melissa (Total: -6.33, Counter: -3.70, Synergy: -2.63)
Phoveus (Total: -6.64, Counter: -6.64, Synergy: 0.00)
Ixia (Total: -6.82, Counter: -3.97, Synergy: -2.85)
Diggie (Total: -8.00, Counter: -8.00, Synergy: 0.00)
```

Gambar 25. Eksperimen Pemilihan Hero ke-3
Sumber: Dokumentasi Pribadi

Pada akhir program ketika kita semuanya sudah memilih maka akan ditampilkan komposisi hero terakhir dari masing-masing tim dan menunjukkan seberapa besar *counter score* dan *compatibility score* dari tiap tim.

```
Final Draft:
Team A: Miya, Natalia, Gloo, Karina, Helcurt
Team B: Belerick, Hanzo, Chip, Wanwan, Thamuz

Final Score:
Team A - Synergy Score: 18.56, Counter Score: 12.19
Team B - Synergy Score: 32.04, Counter Score: -12.19
```

Gambar 26. Hasil Akhir Eksperimen
Sumber : Dokumentasi Pribadi

Perolehan skor akhir menunjukkan bahwa tim B memiliki skor sinergi yang lebih besar dibandingkan tim A. Namun, skor *counter* tim B bernilai negatif sedangkan tim A bernilai positif. Hal ini mengindikasikan bahwa komposisi hero tim A cenderung meng-*counter* hero pada tim B. Sehingga kedua tim cukup imbang, karena tim A dapat mengcounter hero tim B sedangkan hero tim B bersinergi baik satu sama lain sehingga dapat menghasilkan kerja sama antar hero yang lebih baik jika dibandingkan dengan tim A.

VI. Kesimpulan

Dari penelitian ini dapat disimpulkan bahwa pemain game online pada tiap tahunnya meningkat dan Mobile Legends: Bang Bang merupakan salah satu game online yang populer di Indonesia. Dari game tersebut bagian pemilihan hero merupakan salah satu aspek yang penting dalam permainan game, terutama pada mode *ranked.*

Kemudian didapatkan juga bahwa Graf adalah salah satu konsep dasar yang dapat diimplementasikan ke banyak hal. Salah satu jenis dari graf yang dipakai dalam penelitian ini adalah graf berbobot dan graf berarah yang memungkinkan kita untuk menganalisis hubungan antar hero dengan mendalam dan menghasilkan komposisi tim yang baik dengan memperhatikan sinergi tim dan kemampuan untuk meng-*counter* hero musuh.

Selain dari itu terdapat juga API yang tersedia di Web Official Mobile Legend memungkinkan kita untuk mengolah data dari API hero terkait. Namun API ini tidak memberikan data secara lengkap dan diperlukan ekstraksi mendalam dengan library Selenium dan BeautifulSoup untuk melakukan *parsing* struktur HTML.

VII. Evaluasi dan Saran Pengembangan

Pada penelitian ini data yang dipakai hanya berdasarkan Web Official Mobile Legends : Bang Bang yang memiliki data relasi antar satu hero dengan hero lainnya. Dalam tiap page hero di dalam webnya terdapat 2 kategori, yaitu kategori *counter* dan kategori *compatibility* dengan tiap kategori hanya terdapat 5 terbaik dan 5 terburuk pada tiap kategorinya. Hal ini membuat graf tidak menjadi graf lengkap (graf dengan setiap simpul memiliki busur kepada tiap simpul lainnya), sehingga terdapat celah data relasi yang kosong dan dapat dilihat pada beberapa bagian tidak terdapat skor atau nol.

Walaupun algoritma pencarian hero terbaik dan terburuknya sudah berjalan sesuai dengan spesifikasi, namun akan lebih baik lagi jika modal graf yang diterapkan adalah graf lengkap dengan cara mendapatkan data relasi antar hero yang lengkap. Dengan demikian, kalkulasi dari pemilihan hero terbaik dan terburuk dapat berjalan dengan lebih optimal karena setiap hero memiliki hubungan yang jelas dengan hero lainnya.

Selain itu, menurut penulis masih banyak pengembangan yang dapat dilakukan dari algoritmanya, contohnya yaitu untuk memperhatikan *meta hero* dan *win rate* dari tiap hero. Hal ini dapat berguna dalam sesi *first pick,* karena di dalam algoritma ini perbandingan hanya dapat dilakukan setelah ada hero yang dipilih. Sehingga, pada pilihan pertama tidak terdapat rekomendasi akan hero apa yang paling terbaik untuk dipilih. Selain itu jika kita memasukkan konsiderasi *meta hero* dan juga *win rate* maka algoritma pemilihannya juga dapat menjadi lebih matang.

VIII. Lampiran

Sumber kode implementasi dapat dilihat di Github Repository berikut :
https://github.com/ahsuunn/Draft-Pick-ML-Weighted-Graph

IX. Ucapan Terima Kasih

Saya ingin mengucapkan syukur kepada Tuhan YME. karena berkatnya makalah ini dapat selesai dengan baik. Selanjutnya saya ingin berterima kasih kepada pak Rila Mandala selaku dosen mata kuliah Matematika Diskrit IF 1220 yang telah memberikan pemahaman mendalam akan mata kuliah ini, sehingga saya dapat mengerjakan makalah ini dengan pemahaman penuh. Selain itu saya juga ingin berterima kasih kepada rekan saya di jurusan Teknik Informatika ini yang sering mengajak saya bermain Mobile Legend sehingga saya terinspirasi untuk mengaplikasikan ilmu dari Matematika Diskrit pada game tersebut.

Referensi

[1] BeautifulSoup Documentation, “BeautifulSoup Documentation,” Crummy, 2024. https://www.crummy.com/software/BeautifulSoup/bs4/doc/. [Diakses pada 3 Januari 2025].

[2] Liputan6, “Pemain Game di Indonesia Makin Meningkat, Mobile Legends: Bang Bang Jadi Mobile Game Terpopuler,” Liputan6, 2024. https://www.liputan6.com/lifestyle/read/5749764/pemain-game-di-indonesia-makin-meningkat-mobile-legends-bang-bang-jadi-mobile-game-terpopuler. [Diakses pada 4 Januari 2025].

[3] NetworkX Documentation, “NetworkX Documentation,” NetworkX, 2024. https://networkx.org/documentation/stable/reference/index.html. [Diakses pada 4 Januari 2025].

[4] R. Munir, “Matematika Diskrit: Teori Graf Bagian 1,” 2024. https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf. [Diakses pada 4 Januari 2025].

[5] Selenium Documentation, “Selenium Documentation,” Selenium, 2024. https://www.selenium.dev/documentation/. [Diakses pada 4 Januari 2025].

[6] Telkom University, “Teori Graf: Sejarah, Manfaat, dan Aplikasinya,” Telkom University, 2024. https://surabaya.telkomuniversity.ac.id/teori-graf-sejarah-manfaat-dan-aplikasinya/.[Diakses pada 4 Januari 2025].

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 26 Desember 2024

Ahsan Malik Al Farisi - 13523074