Analisis Pemanfaatan Graf Berarah tanpa Siklus dalam Sistem Pemungutan Suara Tideman

Irvin Tandiarrang Sumual - 13524030
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung 40132, Indonesia
13524030@std.stei.itb.ac.id

***Abstrak*—Pemungutan suara merupakan salah satu unsur penting dalam suatu sistem demokrasi, di mana keputusan bersama diambil berdasarkan preferensi individu. Namun, metode pemungutan suara konvensional seringkali menghasilkan pemenang yang tidak mencerminkan keinginan mayoritas secara keseluruhan. Salah satu alternatif untuk mengatasi permasalahan ini ialah pemungutan suara dengan metode Tideman, sebuah sistem pemungutan suara berdasarkan urutan preferensi yang mempertimbangkan semua pasangan kandidat secara *head-to-head*. Metode ini memanfaatkan graf berarah tanpa siklus (*directed acyclic graph*) untuk merepresentasikan hasil perbandingan pasangan kandidat secara *head-to-head*. Melalui penambahan sisi graf secara selektif untuk menghindari terbentuknya suatu siklus, graf yang terbentuk akan konsisten dan memungkinkan identifikasi pemenang secara adil, yaitu kandidat yang tidak pernah dikalahkan oleh kandidat lain secara langsung maupun tidak langsung.**

Kata Kunci*— Graf Berarah, Siklus Graf , Graf Berarah tanpa Siklus (*Directed Acyclic Graph*), *Depth-First Search*, Metode Tideman, *Ranked Pairs

I. Pendahuluan

Pemungutan suara merupakan sarana utama dalam sistem demokrasi untuk menentukan pemimpin secara adil berdasarkan suara rakyat. Akan tetapi, pada kenyataannya, seringkali sistem pemungutan suara yang digunakan tidak mencerminkan keinginan mayoritas secara menyeluruh, misalnya saja pada metode Plurality. Pada metode ini, kandidat yang terpilih bisa saja bukanlah kandidat yang sebenarnya didukung oleh sebagian besar pemilih, tetapi terpilih hanya karena ia memperoleh suara terbanyak dibandingkan masing-masing kandidat lain.

Permasalahan dalam metode-metode pemungutan suara misalnya pada metode Plurality akan mengakibatkan kurangnya rasa adil dan juga ketepatan representasi dalam sistem pemungutan suara. Untuk mengatasi hal tersebut, dibutuhkan metode yang tidak hanya memperhitungkan pilihan utama pemilih, melainkan memperhitungkan urutan peringkat dari pilihan pemilih. Salah satu metode yang berpotensi menjadi jawaban dari permasalahan yang ada ialah metode Tideman atau dikenal juga dengan *Ranked Pairs*.

Metode Tideman menggunakan konsep graf berarah tanpa siklus (*directed acyclic graph*) untuk merepresentasikan hasil perbandingan pasangan kandidat secara *head-to-head*. Hal ini penting agar tidak terjadi kontradiksi dalam preferensi keseluruhan yang mengakibatkan pemenang tidak dapat ditentukan. Setiap sisi (*edge*) pada graf menunjukkan bahwa kandidat tertentu lebih disukai dibandingkan kandidat lainnya. Penentuan pemenang dilakukan dengan menambahkan sisi-sisi secara berurutan berdasarkan kekuatan preferensi pemilih tanpa membentuk suatu siklus sehingga diperoleh kandidat yang tidak pernah dikalahkan oleh kandidat manapun.

Di dalam makalah ini, penulis menganalisis pemanfaatan graf berarah tanpa siklus (*directed acyclic graph*) dalam sistem pemungutan suara dengan metode Tideman. Selain itu, penulis juga membahas mengenai proses pembentukan graf, pendeteksian siklus, dan penentuan pemenang berdasarkan urutan preferensi pemilih.

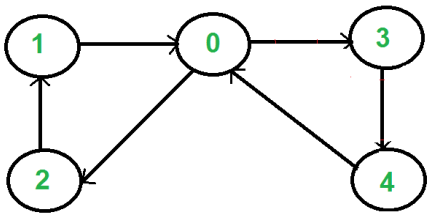
II. Landasan Teori

A. Teori Graf

A.1. Definisi Graf Berarah

Graf merupakan struktur data nonlinear yang terdiri dari simpul (*vertice*) dan sisi (*edge*) [2]. Graf pada umumnya digunakan untuk merepresentasikan objek-objek diskrit beserta hubungan antarobjek yang ada [1]. Secara formal, sebuah graf G didefnisikan sebagai pasangan G = (V, E), di mana V merupakan himpunan tidak kosong yang terdiri dari simpul-simpul, misalnya V = { v_1, v_2, v_3, … , v_n }. Sementara itu, E merupakan himpunan sisi (*edges*) yang menghubungkan sepasang simpul, misalnya E = { e_1, e_2, e_3, … , e_n } [1].

Graf berarah merupakan salah satu jenis dari graf berdasarkan orientasi arahnya [1]. Graf berarah didefinisikan sebagai graf yang setiap sisinya memiliki arah yang telah ditentukan [3].



Gambar 1: Graf berarah (Sumber : [3])

A.2. Karakteristik Graf Berarah

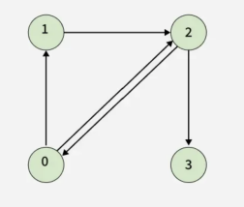
Graf berarah memiliki sejumlah karakteristik[3]:

1. Setiap sisinya merupakan sisi berarah (memiliki arah dari satu simpul ke simpul lainnya yang menyatakan hubungan satu arah).

2. Setiap simpul memiliki derajat masuk dan derajat keluar (jumlah sisi yang masuk ke dan keluar dari simpul tersebut).

3. Lintasan dalam graf berarah mengikuti arah dari sisi-sisi yang ada.

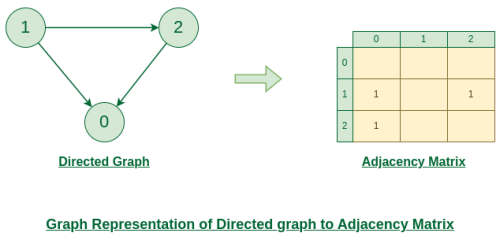
4.Graf berarah dapat mengandung suatu siklus, yaitu lintasan yang berawal dan berakhir pada simpul yang sama tanpa mengulang sisi.



Gambar 2 : Graf yang Mengandung Siklus (0 -> 2 -> 0) (Sumber : [5]).

A.3. Matriks Ketetangaan (Adjacency Matrix)

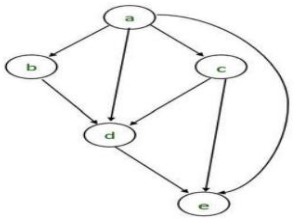
Adjacency Matrix merupakan salah satu carau ntuk merepresentasikan suatu matriks selain representasi dengan matriks bersisian (*incidency matrix*) ataupun senarai ketetangaan (*adjacency list*) [4]. *Adjacency Matrix* merupakan sebuah matriks persegi berukuran *n* x *n*, di mana *n* merupakan jumlah simpul pada suatu graf. Setiap elemen dalam suatu *adjacency matrix* M[i][j] merupakan suatu boolean (0 dan 1) [2]. Pada graf berarah, suatu elemen pada *adjacency matrix* akan bernilai 1 jika terdapat sisi berarah dari simpul v_i ke simpul v_j dan bernilai 0 jika tidak ada sisi tersebut.



Gambar 3 : Representasi Suatu Graf Berarah (Directed Graph) dalam Suatu *Adjacency Matrix* (Sumber: [2]).

A.4. Directed Acyclic Graph

Graf berarah yang tak memiliki siklus merupakan sebuah graf berarah yang tidak mengandung siklus sama sekali.



Gambar 4: Graf Berarah yang Tak Memiliki Siklus (Sumber: [6]).

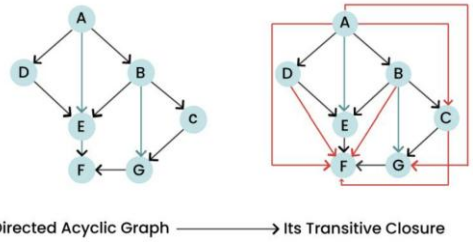
Terdapat dua karakteristik utama dari sebuah graf berarah tanpa siklus (*directed acyclic graph*) [6]:

1. Sisi Berarah.
   Hal ini berarti setiap sisi dalam graf memiliki arah tertentu. Sisi yang menghubungkan satu simpul ke simpul lainnya hanya satu arah saja.

2. Tanpa Siklus
   Hal ini berarti graf tidak mengandung siklus sama sekali. Dengan kata lain, tidak mugkin untuk menelusuri rangkaian sisi berarah dan kembali ke simpul awal mengikuti arah panah. Pembentukan siklus dilarang dalam struktur DAG ini.

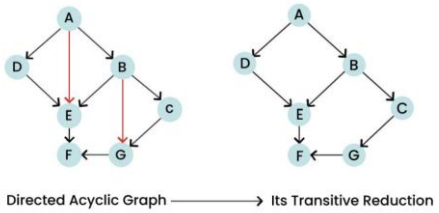
Terdapat sejumlah sifat dari sebuah graf berarah tanpa siklus (*directed acyclic graph*) [6]:

1. *Reachability*
   Dalam DAG, simpul A dikatakan dapat dijangkau dari simpul B jika terdapat lintasan yang dimulai dari B dan berakhir di A.
2. *Transitive Closure*
   Transitive closure akan menunjukkan simpul-simpul mana saja yang dapat dicapai dari simpul lain melalui satu atau lebih sisi berarah.



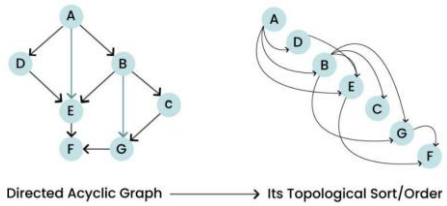
Gambar 5: Sebuah DAG dan *Transitive Closure*-nya (Sumber: [6]).

3. *Transitive Reduction*
   Transitive reduction dari suatu graf berarah merupakan sebuah graf yang hanya mempertahankan hubungan-hubungan langsung antarsimpul dan menghilangkan sisi-sisi yang dapat disimpulkan dari sisi yang tersisa.



Gambar 6: Sebuah DAG dan *Transitive Reduction*-nya (Sumber: [6]).

4. *Topological Ordering*
   Sebuah DAG dapat diurutkan secara topologis artinya simpul-simpul dapat disusun dalam urutan linear sehingga setiap sisi graf mengacu dari sismpul yang muncul lebih awal ke simpul yang muncul akhiran.



Gambar 7: Sebuah DAG dan *Topological Ordering*-nya (Sumber: [6]).

A.5. Algoritma Pendeteksi Siklus pada Graf Berarah

Salah satu pendekatan untuk mendeteksi siklus dalm graf berarah adalah dengan menggunakan algoritma Depth First Search (DFS) [5]. Pada prinsipnya, DFS akan bekerja dengan cara menelusuri simpul-simpul secara mendalam dan jika selama penelusuran ditemukan jalur yang kembali ke simpul sebelumnya, graf mengandung siklus.

Secara umum, langkah-langkah deteksi siklus pada graf berarah dengan DFS ialah sebagai berikut [8]:

1. Menyiapkan dua buah array boolean.
   Satu array digunakan untuk menandai simpul yang telah dikunjungi (*visited*) dan satu lagi untuk melacak simpul yang sedang berada dalam jalur penelusuran saat ini (*recursion stack*).
2. Gunakan pendekatan rekursif untuk mengunjungi semua simpul tetangga yang belum dikunjungi dari simpul saat ini.
3. Jika selama proses penelusuran ditemukan simpul yang telah dikunjung dan masih berada dalam *recursion stack*, dapat disimpulkan terdapat siklus dalam graf
4. Jika suatu simpul tidak memiliki tetangga yang belum dikunjungi, algoritma akan (*backtrack*) ke simpul sebelumnya untuk melanjutkan penelusuran ke jalur lain yang mungkin.

B. Metode Tideman

Metode Tideman merupakan salah satu algoritma pemungutan suara berbasis peringkat yang dirancang untuk memilih pemenang dengan mempertimbangkan seluruh urutan preferensi dari pemilih.

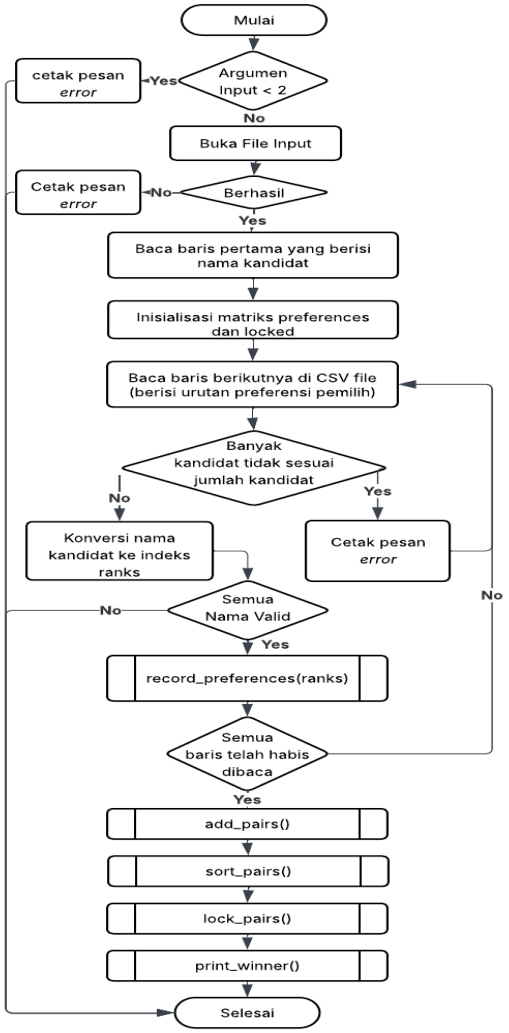
Langkah-langkah utama metode Tideman [7]:

1. Pencatatan Preferensi Masing-Masing Pemilih.
2. Penghitungan jumlah preferensi langsung antara setiap pasangan kandidat. Untuk setiap pasangan kandidat A dan B, sistem akan mencatat berapa banyak pemilih yang lebih memilih A dibandingkan B beserta margin kemenangannya.
3. Penyortiran pasangan kandidat dalam urutan menurun berdasarkan kekuatan kemenangan (selisih jumlah pemilih yang memilih kandidat pemenang atas kandidat yang kalah).
4. Penguncian pasangan yang dimulai dari pasangan dengan margin kemenangan terbesar ke dalam graf berarah jika dan hanya jika penguncian tersebut tidak membentuk suatu siklus. Proses ini dilakukan secara iteratif hingga terbentuk sebuah *directed acyclic graph* yang merupakan hasil dari pemungutan suara dengan metode Tideman.
5. Penentuan pemenang dengan memilih kandidat yang tidak memiliki sisi masuk.

III. PEMBAHASAN

Penulis telah membuat program yang dapat menyimulasikan pemilihan umum dengan metode Tideman. Program yang penulis buat merupakan suatu program dalam bahasa Python yang merupakan hasil pengembangan dan translasi dari program yang pernah penulis buat dalam bahasa C untuk menjawab permasalahan di [7]. Dalam proses mengubah ke bahasa Python, penulis menggunakan bantuan dari DeepSeek dan ChatGPT.

A. Alur Utama Program



Gambar 8 : Alur Program yang Digunakan untuk Menyimulasikan pemungutan suara dengan Metode Tideman (Sumber: Arsip Penulis).

B. Format Masukan Program

Program yang telah dibuat akan menerima preferensi dari masing-masing pemilih yang telah dikumpul dalam suatu CSV file, di mana pada baris ke-1 di CSV file merupakan nama-nama kandidat yang valid lalu diikuti dengan preferensi masing-masing pemilih terhadap kandidat dengan menempatkan kandidat yang paling disukai di paling kiri sampai dengan yang paling kurang disukai di sebelah kanan.

```
Alice,Bob,Charlie
Alice,Bob,Charlie
Alice,Bob,Charlie
Alice,Bob,Charlie
Bob,Charlie,Alice
Bob,Charlie,Alice
Charlie,Alice,Bob
Charlie,Alice,Bob
Charlie,Alice,Bob
Charlie,Alice,Bob
```

Gambar 9 : Contoh File CSV yang Menunjukkan Kandidat Berjumlah 3 Orang (Alice, Bob, Charlie) Disertai dengan Preferensi Masing-Masing Pemilih yang Berjumlah 9 Pemilih (Sumber: Arsip Penulis).

C. Representasi Preferensi Pemilih dalam Program

Preferensi pemilih dalam program direpresentasikan melalui struktur data berupa matriks dua dimensi bernama preferences. Matriks ini berukuran *n* x *n*, di mana *n* merupakan jumlah kandidat yang ada. Setiap elemen preferences[i][j] akan merepresentasikan jumlah pemilih yang lebih memilih kandidat ke-i dibandingkan kandidat ke-j.

Matriks preferences akan diperbarui setiap program memproses satu baris masukan dari pemilih (dari suatu CSV file). Data urutan preferensi dari pemilih akan dikonversi menjadi array ranks yang berisi indeks kandidat berdasarkan peringkat. Setelah itu, fungsi record_preferences(ranks) akan membandingkan setiap pasangan kandidat dalam array tersebut. Untuk setiap pasangan kandidat (i,j) di mana kandidat i muncul sebelum kandidat j di array ranks, nilai preferences[i][j] akan ditambah satu.

```
# Update global preferences matrix based on a single voter's ranks
def record_preferences(ranks):
    for i in range(candidate_count):
        found = False
        for j in range(candidate_count):
            if ranks[j] == i:
                found = True
            elif found:  # If i was found earlier, i is preferred over ranks[j]
                preferences[i][ranks[j]] += 1
```

Gambar 10 : Fungsi record_preferences(ranks) (Sumber: Arsip Penulis).

Sebagai ilustrasi, jika terdapat tiga kandidat dan seorang pemilih memberikan preferensi dengan urutan [2, 0, 1], program akan memperbarui nilai preferences dengan menambahkan 1 ke preferences[2][0], preferences[2][1], dan preferences[0][1] yang menunjukkan bahwa pemilih lebih menyukai kandidat 2 dibandingkan 0 maupun 1 dan lebih menyukai 0 dibandingkan 1.

D. Pembentukan dan Pengurutan Pasangan Kandidat

Setelah seluruh preferensi pemilih telah dicatat ke matriks preferences, langkah selanjutnya ialah membentuk pasangan kandidat berdasarkan hasil preferensi tersebut. Setiap pasangan terdiri dari dua kandidat, di mana satu kandidat lebih disukai dibandingkan kandidat lainnya.

Pembentukan pasangan akan dilakukan dengan menggunakan fungsi add_pairs() yang bertugas untuk membandingkan setiap pasangan kandidat (i,j) yang berbeda. Jika jumlah pemilih yang menyukai kandidat i lebih banyak dibandingkan kandidat j, maka pasangan yang ditambahkan ialah (i,j) di mana i menang, sedangkan j kalah, begitu juga sebaliknya.

```
# Add all pairs of candidates where one is preferred over the other
def add_pairs():
    global pair_count
    for i in range(candidate_count):
        for j in range(i + 1, candidate_count):
            if preferences[i][j] > preferences[j][i]:
                pairs.append(Pair(i, j))
                pair_count += 1
            elif preferences[j][i] > preferences[i][j]:
                pairs.append(Pair(j, i))
                pair_count += 1
```

Gambar 11 : Fungsi add_pairs() (Sumber: Arsip Penulis).

Setelah semua pasangan terbentuk, program akan mengurutkan pasangan-pasangan tersebut berdasarkan kekuatan kemenangan (selisih jumlah pemilih yang menyukai pemenang dengan yang menyukai yang kalah). Pengurutan dilakukan secara menurun dengan menggunakan fungsi sort_pairs().

```
# Sort the pairs in decreasing order of strength of victory
def sort_pairs():
    def get_strength(pair):
        return preferences[pair.winner][pair.loser]
    pairs.sort(key=get_strength, reverse=True)
```

Gambar 12 : Fungsi sort_pairs() (Sumber: Arsip Penulis)

Tahap pembentukan dan pengurutan pasangan ini merupakan penghubung antara data preferensi pemilih dengan representasi graf yang akan dibentuk dalam menentukan proses menentukan pemenang suatu pemungutan suara dengan metode Tideman. Tahapan ini memastikan bahwa urutan menambahkan sisi ke dalam graf akan dilakukan dengan semestinya.

E. Proses Pembentukan Graf Kandidat

Tahap ini merupakan proses pembentukan graf berarah tanpa siklus dari pasangan-pasangan kandidat yang telah diurutkan berdasarkan kekuatan kemenangan di list pairs. Proses ini disebut juga dengan istilah *locking pairs*, di mana setiap pasangan kandidat diwakili sebagai sisi (*edge*) dari simpul pemenang menuju simpul yang kalah.

Tujuan utama dari tahap ini ialah membentuk graf yang menggambarkan urutan preferensi keseluruhan pemilih tanpa membentuk siklus. Siklus dalam kasus ini perlu dihindari

karena jika terbentuk siklus maka tidak akan ada pemenang yang valid dikarenakan jika A lebih disukai dari B, B lebih disukai dari C, tidak mungkin C lebih disukai dari A jika diinginkan seorang pemenang. Oleh karena itu, setiap pasangan kandidat akan di-*lock* terlebih dahulu lalu dianalisis oleh program terkait dampak dari penambahan sisi yang dilakukan. Jika penambahan sisi mengakibatkan suatu siklus terbentuk, pasangan tersebut akan tidak jadi di-*lock.*

Setiap pasangan yang berhasil di-*lock* tanpa menciptakan siklus akan tetap dimasukkan ke matriks locked, yaitu sebuah *adjacency matrix* dari graf berarah. Dalam matriks ini, untuk setiap posisi [i][j] akan menunjukkan adanya sisi dari kandidat i ke kandidat j jika elemen pada posisi tersebut bernilai True, berlaku sebaliknya.

```
# Lock pairs into the candidate graph without creating cycle
def lock_pairs():
    for pair in pairs:
        locked[pair.winner][pair.loser] = True
        if path_exists(pair.loser, pair.winner):
            locked[pair.winner][pair.loser] = False
```

Gambar 13: Fungsi lock_pairs() yang Digunakan untuk Membentuk Graf Berarah yang Merepresentasikan Hasil Suatu Pemungutan Suara (Sumber: Arsip Penulis).

Seiring bertambahnya pasangan yang di-*lock*, graf perlahan akan terbentuk menjadi sebuah *directed acyclic graph* (DAG). Melalui DAG ini, pemenang dapat ditentukan dengan cara mencari simpul yang tidak memiliki sisi masuk (*indegree*) yang merepresentasikan kandidat yang tidak dikalahkan oleh kandidat manapun secara langsung maupun tidak langsung.

F. Pendeteksian Siklus dalam Graf

Dalam proses pembentukan graf kandidat, program perlu melakukan proses pendeteksian siklus secara dinamis untuk mencegah terbentuknya suatu siklus. Terdapat beragam algoritma yang dapat dilakukan untuk mendeteksi apakah dengan menambahkan suatu sisi akan membuat graf memiliki siklus atau tidak. Dalam program yang dibuat, algoritma yang digunakan ialah algoritma *Depth-First Search* (DFS).

Pendeteksian siklus dalam graf ini akan dibantu dengan menggunakan fungsi path_exists. Fungsi ini bekerja secara rekursif menelusuri lintasan dari satu kandidat ke kandidat lainnya berdasarkan matriks locked. Jika saat penelusuran ditemukan bahwa kandidat tujuan sudah pernah dikunjungi dalam lintasan yang sedang dilalui maka hal tersebut menandakan adanya siklus ketika penambahan sisi dilakukan. Dengan demikian, penambahan sisi yang baru dilakukan akan dibatalkan untuk mencegah terbentuknya siklus. Pendekatan yang dilakukan ini memastikan bahwa graf yang terbentuk tetap merupakan *Directed Acyclic Graph* (DAG) sehingga pemenang suatu pemungutan suara tetap dapat ditentukan.

```
# Recursive function to check whether a path exists
# from one candidate to another (to avoid cycles)
def path_exists(from_cand, to_cand, visited=None):
    if visited is None:
        visited = set()

    if to_cand in visited:
        return True

    visited.add(from_cand)

    for i in range(candidate_count):
        if locked[i][from_cand]:
            if path_exists(i, to_cand, visited.copy()):
                return True
    return False
```

Gambar 14: Fungsi path_exists(from_cand, to_cand, visited = None) (Sumber: Arsip Penulis).

G. Simulasi dan Visualisasi Hasil dari Pemungutan Suara dengan Metode Tideman

Dalam simulasi yang dilakukan, terdapat 15 orang pemilih untuk 5 kandidat, yaitu Doraemon, Nobita, Shizuka, Giant, Suneo.

```
1   Doraemon,Nobita,Shizuka,Giant,Suneo
2   Doraemon,Nobita,Shizuka,Giant,Suneo
3   Doraemon,Nobita,Shizuka,Giant,Suneo
4   Doraemon,Nobita,Shizuka,Giant,Suneo
5   Doraemon,Nobita,Shizuka,Giant,Suneo
6   Doraemon,Nobita,Shizuka,Giant,Suneo
7   Nobita,Shizuka,Giant,Suneo,Doraemon
8   Nobita,Shizuka,Giant,Suneo,Doraemon
9   Nobita,Shizuka,Giant,Suneo,Doraemon
10  Nobita,Shizuka,Giant,Suneo,Doraemon
11  Nobita,Shizuka,Giant,Suneo,Doraemon
12  Nobita,Shizuka,Giant,Suneo,Doraemon
13  Shizuka,Giant,Suneo,Doraemon,Nobita
14  Shizuka,Giant,Suneo,Doraemon,Nobita
15  Shizuka,Giant,Suneo,Doraemon,Nobita
16  Shizuka,Giant,Suneo,Doraemon,Nobita
```

Gambar 15: Nama Kandidat (baris ke-1) dan Preferensi Masing-Masing Pemilih dalam Suatu CSV File (baris ke-2 sampai baris ke-16) (Sumber: Arsip Penulis).

```
Preferences Matrix:
          Doraemon  Nobita  Shizuka  Giant  Suneo
Doraemon         0       9        5      5      5
Nobita           6       0       11     11     11
Shizuka         10       4        0     15     15
Giant           10       4        0      0     15
Suneo           10       4        0      0      0
```

Gambar 16: Matriks yang Berisi Preferensi Pemilih (Matriks Preference) (Sumber: Arsip Penulis).

```
Pairs (before sorting):
Doraemon beats Nobita with margin 3
Shizuka beats Doraemon with margin 5
Giant beats Doraemon with margin 5
Suneo beats Doraemon with margin 5
Nobita beats Shizuka with margin 7
Nobita beats Giant with margin 7
Nobita beats Suneo with margin 7
Shizuka beats Giant with margin 15
Shizuka beats Suneo with margin 15
Giant beats Suneo with margin 15
```

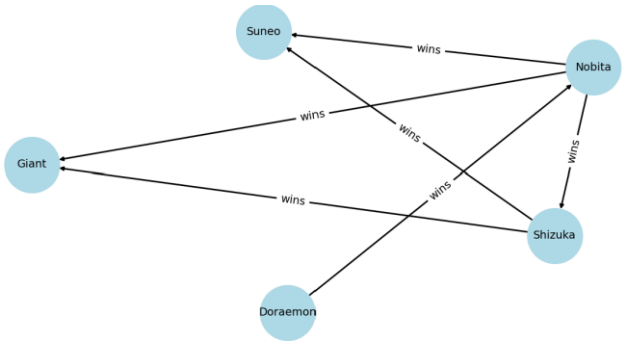
Gambar 17: List Pairs sebelum Diurutkan (Sumber: Arsip Penulis).

```
Pairs (after sorting):
Shizuka > Giant (margin: 15)
Shizuka > Suneo (margin: 15)
Giant > Suneo (margin: 15)
Nobita > Shizuka (margin: 7)
Nobita > Giant (margin: 7)
Nobita > Suneo (margin: 7)
Shizuka > Doraemon (margin: 5)
Giant > Doraemon (margin: 5)
Suneo > Doraemon (margin: 5)
Doraemon > Nobita (margin: 3)
```

Gambar 18: List Pairs setelah Diurutkan (Sumber: Arsip Penulis)

```
Locked Matrix:
          Doraemon  Nobita   Shizuka   Giant     Suneo
Doraemon  F         T        F         F         F
Nobita    F         F        T         T         T
Shizuka   F         F        F         T         T
Giant     F         F        F         F         F
Suneo     F         F        F         F         F
```

Gambar 19 : Matris Locked (Sumber: Arsip Penulis)



Gambar 20: Graf Berarah tanpa Siklus yang Terbentuk (Sumber: Arsip Penulis)

Berdasarkan graf berarah yang terbentuk, pemenang dari pemungutan suara ini ialah Doraemon dikarenakan hanya Doraemon yang memiliki *indegree* bernilai 0.

H. Analisis Kasus Siklus Condorcet Seimbang

Dalam beberapa konfigurasi pemungutan suara, dapat terjadi situasi di mana preferensi pemilih membentuk siklus Condorcet yang seimbang, yaitu kondisi ketika setiap kandidat menang dari satu kandidat lain, tetapi juga kalah dari kandidat yang berbeda sehingga tidak ada pemenang mutlak.

Contoh sederhana dari kondisi ini ialah ketika terdapat 9 pemilih dan 3 kandidat dengan distribusi suara sebagai berikut:

- 3 pemilih memilih A > B > C
- 3 pemilih memilih B > C > A
- 3 pemilih memilih C > A > B

Berdasarkan preferensi tersebut, perbandingan *head-to-head* menunjukkan bahwa:

- A mengalahkan B (6 suara melawan 3 suara)
- B mengalahkan C (6 suara melawan 3 suara)
- C mengalahkan A (6 suara melawan 3 suara)

Seharusnya, graf yang terbentuk akan memiliki siklus yang urutannya juga bergantung pada urutan *locking pairs* yang dilakukan. Dalam kasus ini, umumnya graf berarah yang terbentuk ialah A > B > C > A ataupun C > B > A > C. Meskipun metode Tideman dirancang untuk menghindari pembentukan siklus dalam penyusunan graf, hasil akhirnya akan sangat bergantung pada *locking* pairs saja. Dalam kasus ini, dikarenakan semua margin kemenangan sama besar (selisih 3 suara) maka urutan dalam *locking* bisa berbeda tergantung dengan implementasi yang dilakukan sehingga hasil akhir dapat berbeda, misalnya dalam kasus ini bisa saja ada implementasi yang menghasilkan C sebagai pemenang, sedangkan implementasi lainnya menghasilkan A sebagai pemenang.

Dengan demikian, meskipun metode Tideman menjamin adanya satu pemenang (karena graf yang terbentuk selalu *directed acyclic graph*), dalam kasus ini hasilnya akan acak. Hal ini tentunya tidak diinginkan. Oleh karena itu, tetap diperlukan evaluasi terhadap hasil pemungutan suara menggunakan metode Tideman dalam kasus seperti ini.

I. Pemanfaatan Graf Berarah tanpa Siklus dalam Metode Tideman

Di dalam metode Tideman, graf yang digunakan bukanlah sekadar graf berarah biasa, tetapi secara spesifik menggunakan graf berarah tanpa siklus. (*Directed Acyclic Graph*). Graf berarah tanpa siklus ini terbentuk selama proses *locking pairs*, yaitu proses ketika pasangan kandidat ditambahkan sebagai sisi dalam graf jika dan hanya jika penambahan tersebut tidak mengakibatkan terbentuknya suatu siklus. Berikut ini sejumlah pemanfaatan graf berarah tanpa siklus dalam metode tideman:

1. Representasi relasi preferensi antarkandidat.
   Graf berarah tanpa siklus digunakan untuk menggambarkan hubungan dominasi antarkandidat berdasarkan mayoritas pemilih.
2. Menjamin konsistensi preferensi keseluruhan.
   Graf berarah tanpa siklus menjaga agar tidak terbentuk siklus sehingga tidak terjadi kontradiksi dalam hasil, misalnya A menang dari B, B menang dari C, tetapi C menang dari A. Hal ini dapat terjadi karena penambahan sisi dalam graf selalu diperiksa untuk memastikan bahwa penambahan sisi tidak akan mengakibatkan terbentuknya suatu siklus.
3. Dasar penentuan pemenang.
   Ketika graf berarah tanpa siklus telah terbentuk, pemenang dapat ditentukan dengan hanya melihat simpul yang tidak memiliki sisi masuk (indegree = 0).

Graf berarah tanpa siklus memiliki sifat penting, yaitu *topological* order yang dapat dimanfaatkan untuk menemukan kandidat yang tidak pernah terkalahkan secara langsung maupun tidak langsung. Dengan demikian, graf berarah tanpa siklus dimanfaatkan tidak hanya sebagai representasi visual, tetapi sebagai fondasi utama untuk menentukan pemenang dari suatu pemungutan suara.

IV. KESIMPULAN

Metode Tideman merupakan salah satu metode pemungutan suara berdasarkan peringkat yang mampu menangani preferensi pemilih secara menyeluruh dan konsisten. Dalam implementaasinya, metode ini memanfaatkan graf berarah tanpa siklus (*directed acyclic graph*) untuk

merepresentasikan hasil perbandingan pasangan kandidat berdasarkan preferensi pemilih dan juga sebagai fondasi utama untuk menentukan pemenang dari pemungutan suara yang dilakukan. Dengan membentuk pasangan kandidat yang diurutkan berdasarkan kekuatan kemenangan dan hanya menambahkan sisi ke graf jika tidak mengakibatkan terbentuknya siklus, metode Tideman mampu menjaga konsistensi logika preferensi keseluruhan. Graf berarah tanpa siklus tidak hanya menghindari kontradiksi dalam urutan preferensi, tetapi membantu juga dalam proses penentuan pemenang, yakni dengan mencari simpul yang tidak memiliki sisi masuk (*indegree* = 0).

V. Lampiran

Source code yang digunakan untuk menyimulasikan pemungutan suara dengan metode Tideman :

https://github.com/Irvin-Tandiarrang-Sumual/Analisis-Pemanfaatan-Graf-Berarah-tanpa-Siklus-dalam-Sistem-Pemungutan-Suara-Tideman

VI. Ucapan Terima Kasih

Puji syukur penulis panjatkan kepada Tuhan Yesus yang telah memberikan karunia sehingga penulis dapat menyelesaikan makalah yang berjudul *Analisis Pemanfaatan Graf Berarah tanpa Siklus dalam Sistem Pemungutan Suara Tideman* yang selesai tepat pada waktunya. Penulis juga mengucapkan terima kasih kepada Bapak Arrival Dwi Sentosa, M.T sebagai dosen pengampu mata kuliah IF1220 Matematika Diskrit Kelas 2 atas ilmu dan pengalaman yang dibagikan selama kelas Matematika Diskrit ini. Penulis juga mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir, M.T. sebagai salah satu dosen pengampu Matematika Diskrit yang membagikan banyak sumber pembelajaran Matematika Diskrit di *website* beliau. Penulis juga mengucapkan terima kasih kepada course CS50x [7] yang menjadi inspirasi penulis dan mengenalkan penulis dengan metode Tideman dalam pemungutan suara. Penulis juga mengucapkan terima kasih kepada orang tua, keluarga, teman-teman, dan seluruh pihak yang telah membantu dan mendukung penulis dalam menyelesaikan makalah ini.

Referensi

[1] Rinaldi Munir, "Graf (Bagian 1)", https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/20-Graf-Bagian1-2024.pdf, diakses 17 Juni 2025 pukul 09.36

[2] GeeksforGeeks, "Graph and its representations", https://www.geeksforgeeks.org/dsa/graph-and-its-representations/, diakses 17 Juni 2025 pukul 09.39.

[3] GeeksforGeeks, "What is Directed Graph? | Directed Graph Meaning", https://www.geeksforgeeks.org/dsa/what-is-directed-graph-directed-graph-meaning/, diakses 17 Juni 2025 pukul 10.03.

[4] Rinaldi Munir, "Graf (Bag. 2)", https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2024-2025/21-Graf-Bagian2-2024.pdf, diakses 17 Juni 2025 pukul 10.32

[5] GeeksforGeeks, "Detect Cycle in a Graph", https://www.geeksforgeeks.org/dsa/detect-cycle-in-a-graph/, diakses 17 Juni 2025 pukul 11.47.

[6] GeeksforGeeks, "Introduction to Directed Acyclic Graph (DAG)", https://www.geeksforgeeks.org/dsa/introduction-to-directed-acyclic-graph/, diakses 17 Juni 2025, pukul 11.55.

[7] CS50, "Tideman," https://cs50.harvard.edu/x/2025/psets/3/tideman/, diakses 17 Juni 2025 pukul 12.35

[8] GeeksforGeeks, "Depth First Search (DFS) on a Directed Graph", https://www.geeksforgeeks.org/depth-first-search-or-dfs-on-directed-graph/, diakses 18 Juni 2025, pukul 22.51.

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, 20 Juni 2025

Irvin Tandiarrang Sumual 13524030