Penerapan Teori Graf dan Rekursi dalam Penyelesaian Permaian Sudoku

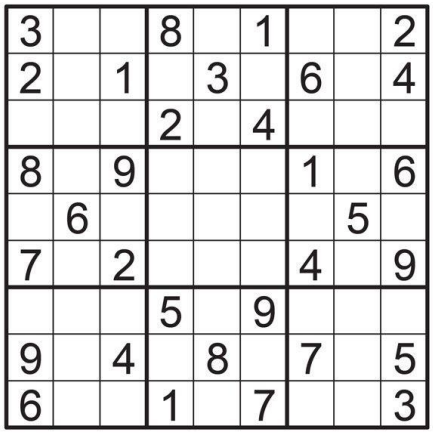
Samy Muhammad Haikal - 13522151[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13522151@itb.ac.id

***Abstract*—Sudoku adalah teka-teki logika yang populer yang membutuhkan pemecahan melalui pengisian angka pada grid 9x9. Tujuan dari permainan ini adalah mengisi setiap kotak dengan angka 1 hingga 9 sedemikian rupa sehingga setiap baris, kolom, dan setiap kotak 3x3 tidak memiliki angka yang sama. Permasalahan ini dapat diselesaikan dengan menggunakan teori pewarnaan graf. Pewarnaan graf akan mewarnai suatu graf dengan warna paling sedikit sehingga tidak ada simpul bertetangga yang berwarna sama. Permasalahan pada Sudoku dapat dimodelkan dan diselesaikan menggunakan teori graf.**

***Keywords*—sudoku, graf, pewarnaan graf, bilangan kromatik, rekursi**

I. PENDAHULUAN

Sudoku adalah teka-teki logika yang terkenal di seluruh dunia karena menantang pemainnya untuk mengisi angka dari 1 hingga 9 ke dalam grid 9x9 yang terbagi menjadi 9 blok 3x3. Setiap baris, kolom, dan blok 3x3 harus diisi sedemikian rupa sehingga tidak ada angka yang berulang dalam setiap baris, kolom, atau blok tersebut. Penyelesaian Sudoku membutuhkan kombinasi keterampilan logis, analitis, dan strategis.



Gambar 1. Contoh Permainan Sudoku

Sumber : https://www.britannica.com/story/will-we-ever-run-out-of-sudoku-puzzles

Berbagai metode dan strategi telah dikembangkan untuk menyelesaikan Sudoku. Salah satu pendekatan yang menarik adalah memanfaatkan konsep teori graf. Teori graf merupakan bagian penting dalam matematika diskrit yang mempelajari interaksi antara objek yang disebut simpul, yang terhubung melalui sisi atau tepi.

Dalam konteks Sudoku, penerapan teori graf melibatkan pemetaan relasi antar angka dalam teka-teki sebagai simpul-simpul dan keterkaitan di antara mereka sebagai sisi-sisi dalam graf. Pendekatan ini memungkinkan penggunaan algoritma graf untuk menganalisis dan menyelesaikan teka-teki Sudoku secara lebih sistematis dan efisien..

II. LANDASAN TEORI

A. Sudoku

Sudoku merupakan sebuah permainan teka-teki logika yang bertujuan mengisi angka-angka dari 1 hingga 9 pada grid 9×9 yang terbagi menjadi sembilan blok 3×3. Tantangannya adalah memastikan bahwa tidak ada angka yang berulang dalam satu baris, satu kolom, atau satu blok. Permainan ini pertama kali muncul di sebuah surat kabar Prancis pada tahun 1895 dan mungkin terinspirasi oleh karya matematikawan Swiss Leonhard Euler, yang dikenal karena karyanya tentang Latin square.

B. Graf

1. Definisi Graf

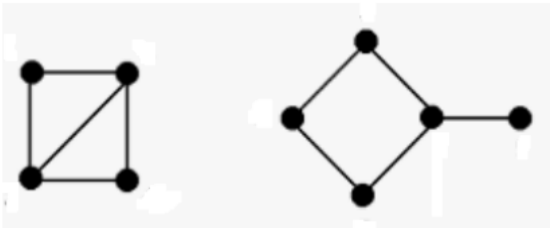
Graf Dalam matematika dan ilmu komputer, sebuah graf (atau grafik) adalah struktur abstrak yang terdiri dari himpunan objek yang disebut simpul atau node, yang terhubung oleh himpunan pasangan terurut yang disebut sisi atau edge. Graf secara visual direpresentasikan sebagai diagram yang terdiri dari simpul-simpul (titik) yang dihubungkan oleh sisi-sisi (garis atau busur). Graf disajikan dalam notasi G = (V,E) dengan V melambangkan himpunan sisi dan E melambangkan himpunan simpul pada graf.

2. Jenis-jenis Graf

Graf dapat dikelompokkan menjadi beberapa kategori berdasarkan karakteristiknya, berikut adalah beberapa jenis graf:

a. Graf tidak berarah

Graf tidak berarah adalah graf yang sisi-sisinya tidak memiliki arah. Dalam hal ini misal graf memiliki simpul {a,b,c,d}, maka sisi {a,b} dan sisi {b,a} dianggap sebagai sisi yang sama.

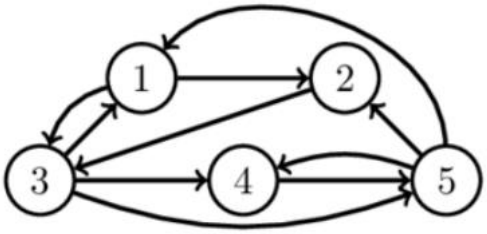


Gambar 2.1 Graf tak berarah

Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf

b. Graf berarah

Graf berarah adalah graf yang sisi-sisinya memiliki arah. Dalam hal ini missal graf memiliki simpul {a,b,c,d}, maka sisi {a,b} dan {b,a} sianggap sebagai dua simpul yang berbeda.

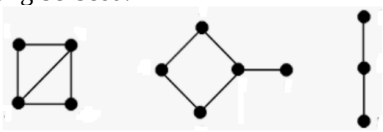


Gambar 2.2 Graf berarah

Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf

c. Graf sederhana

Dalam graf sederhana tidak ada sisi yang menghubungkan simpul dengan dirinya sendiri (self-loop) dan tidak ada lebih dari satu sisi yang menghubungkan dua simpul yang sama. Ini berarti setiap sisi dalam graf sederhana hanya menghubungkan dua simpul yang berbeda.

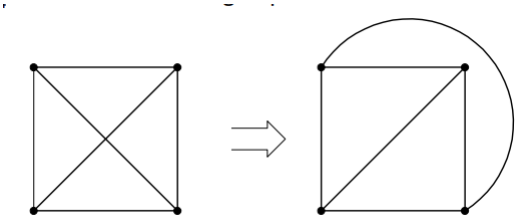


Gambar 2.3 Gambar Graf sederhana

Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf

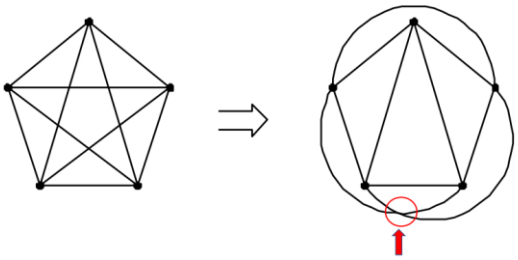
d. Graf planar

Graf planar adalah jenis graf yang dapat digambar di bidang tanpa terjadi tumpang tindih (overlap) pada sisi-sisinya, artinya tidak ada sisi yang bersilangan atau saling memotong ketika graf tersebut digambar dalam bidang dua dimensi.



Gambar 2.4 Graf planar

Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/20-Graf-Bagian2-2023.pdf



Gambar 2.5 contoh graf tidak planar

Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/20-Graf-Bagian2-2023.pdf

e. Graf teratur

Graf teratur adalah sebuah jenis graf yang setiap simpulnya mempunyai derajat yang sama.

3. Terminologi Graf

a. Ketetanggaan

Dalam representasi graf, dua simpul yang terhubung oleh sisi disebut sebagai tetangga satu sama lain. Sebagai contoh, jika terdapat sisi yang menghubungkan simpul A dan simpul B dalam graf, maka A dan B dianggap sebagai tetangga.

b. Derajat

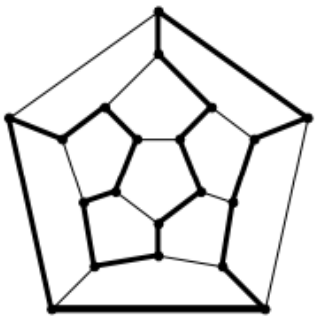
Derajat suatu simpul menyatakan jumlah simpul yang bersisan dengan simpul atau jumlah sisi yang terhubung dengan simpul tersebut

c. Lintasan

Lintasan dalam graf merujuk pada urutan simpul-simpul yang berbeda yang terhubung oleh sisi-sisi graf. Lintasan ini bisa berupa urutan simpul yang saling terhubung oleh sisi secara berurutan dalam graf.

d. Sirkuit

Sirkuit dalam graf merujuk pada lintasan tertutup yang membentuk suatu pola sirkular dengan dimulai dan berakhir di simpul yang sama.

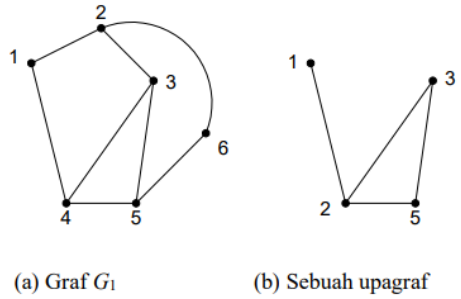


Gambar 2.6 Sirkuit pada Graf

Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/21-Graf-Bagian3-2023.pdf

e. Upagraf

Upagraf adalah sebagian dari suatu graf yang terdiri dari himpunan simpul dan himpunan sisi yang merupakan bagian dari graf yang lebih besar. Dengan kata lain, upagraf adalah graf yang terbentuk dari himpunan simpul dan himpunan sisi yang merupakan bagian dari graf utama.



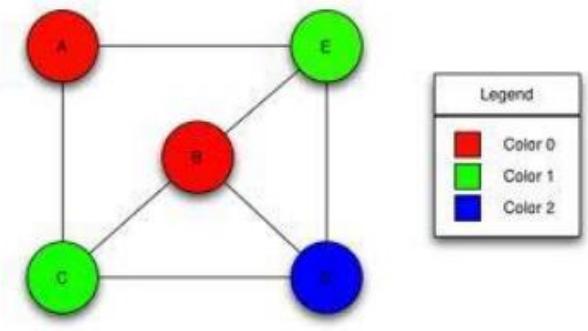
(a) Graf G_1 (b) Sebuah upagraf

Gambar 2.7 Ilustrasi Upagraf pad G1

Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/21-Graf-Bagian3-2023.pdf

C. Pewarnaan Graf

Pewarnaan pada graf adalah proses pemberian warna pada tiap simpul dari graf sehingga tidak ada simpul bertetangga yang memiliki warna yang sama.



Gambar 2.8 Pewarnaan Graf pada graf dengan bilangan kromatik 3

Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/21-Graf-Bagian3-2023.pdf

Bilangan kromatik dari suatu graf adalah jumlah warna minimum yang diperlukan untuk melakukan pewarnaan pada graf. Bilangan kromatik dilambangkan dengan $\chi(G)$.

D. Algoritma Greedy

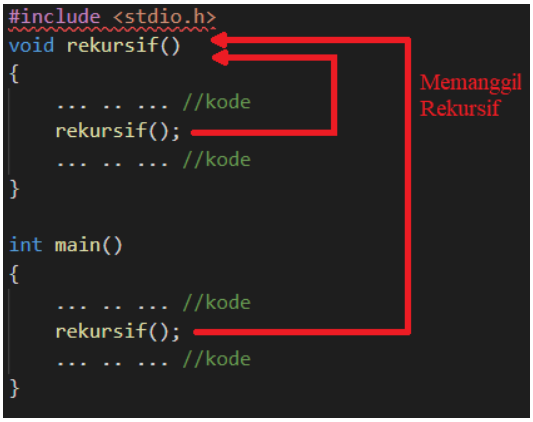
Algoritma greedy adalah algoritma yang mengikuti heuristik pemecahan masalah untuk membuat pilihan optimal secara lokal pada setiap tahap. Algoritma ini akan selalu memilih opsi yang paling menguntungkan saat ini tanpa mempertimbangkan konsekuensi di masa depan.

Algoritma greedy memiliki keunggulan tersendiri karena mudah diimplementasikan dan memiliki waktu komputasi yang cepat. Namun, karena kita tidak mempertimbangkan konsekuensi pilihan kita algoritma greedy bisa saja menghasilkan hasil yang tidak optimal dalam beberapa kasus.

Beberapa contoh algoritma greedy dalam konteks graf adalah algoritma Welsh-Powell, algoritma Prim, dan algoritma Kruskal.

D. Rekursi

Rekursi adalah konsep dalam pemrograman atau matematika yang mengacu pada teknik di mana suatu fungsi atau prosedur memanggil dirinya sendiri secara berulang dalam rangkaian tugas atau perhitungan. Dalam konteks pemrograman, rekursi adalah pendekatan yang memungkinkan fungsi untuk memecah masalah menjadi submasalah yang lebih kecil dan kemudian menyelesaikan setiap submasalah tersebut dengan cara yang sama, secara berulang, hingga mencapai kasus dasar atau terminasi.



Gambar 2.9 Contoh fungsi rekursif

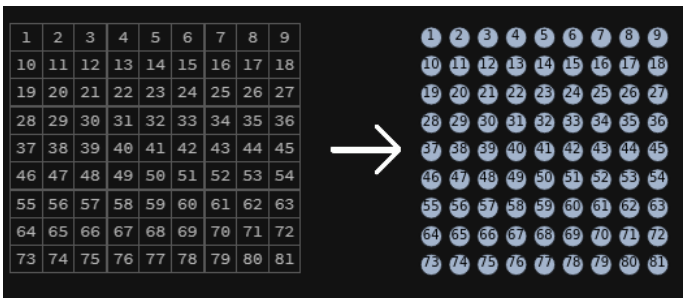
Sumber : https://www.sobatambisius.com/2021/09/belajar-bahasa-c-9-rekursif.html

Fungsi rekursif memberikan beberapa keuntungan yang penting dalam pemodelan, analisis, dan manipulasi struktur data seperti graf dan pohon. Dengan algoritma yang rekursif kita bisa mendapatkan hasil yang lebih optimal disbanding dengan algoritma greedy, tetapi waktu pengeksekusiannya juga akan semakin lambat.

III. IMPLEMENTASI

A. Pemodelan Graf dari Sudoku

Dalam konteks graf, kita bisa memodelkan sebuah puzzle sudoku pada sebuah graf dengan setiap simpul menandakan kotak -kotak yang ada pada papan sudoku.



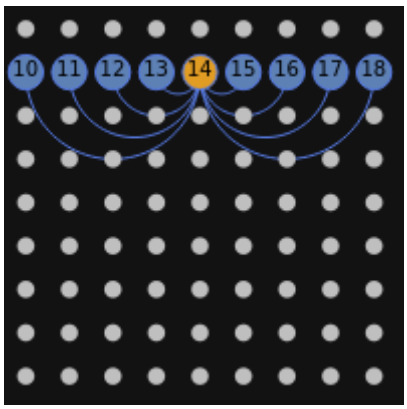
Gambar 3.1 Pemodelan Sudoku menjadi simpul pada graf

Sumber : https://community.wolfram.com/groups/-/m/t/2983903

Untuk memodelkan aturannya kita bisa membaginya menjadi tiga bagian yaitu

1. Tidak boleh ada angka yang berulang pada setiap baris
2. Tidak boleh ada angka yang berulang pada setiap kolom
3. Tidak boleh ada angka yang berulang pada setiap blok 3x3

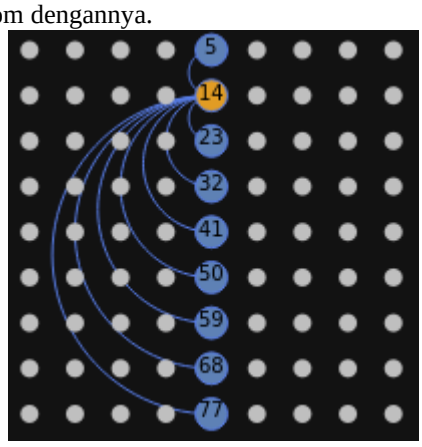
Pertama-tama kita bisa memodelkan aturan 1 dengan menyambungkan tiap simpul dengan simpul yang sebaris dengannya.



Gambar 3.2 Pemodelan aturan 1 pada graf

Sumber : https://community.wolfram.com/groups/-/m/t/2983903

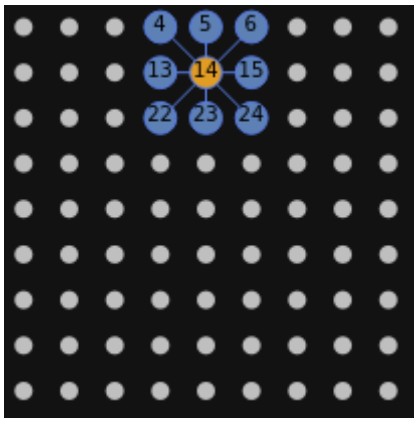
Untuk aturan kedua sambungkan setiap simpul dengan simpul yang sekolom dengannya.



Gambar 3.3 Pemodelan aturan 2 dengan graf

Sumber : https://community.wolfram.com/groups/-/m/t/2983903

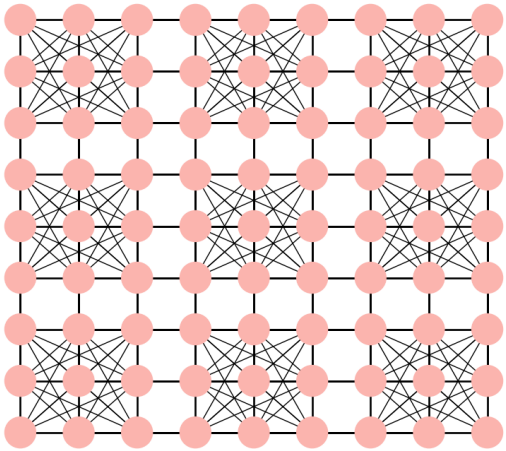
Lalu untuk aturan ketiga, sambungkan setiap simpul dengan simpl yang berada pada blok 3x3 yang sama dengannya.



Gambar 3.3 Pemodelan aturan 3 pada graf

Sumber : https://community.wolfram.com/groups/-/m/t/2983903

Jika aturan-aturan ini sudah dimodelkan maka graf akan berbentuk sebagai berikut. Graf yang terbentuk adalah graf teratur dengan jumlah derajat tiap simpul sama dengan 20.



Gambar 3.4 Pemodelan Sudoku pada graf

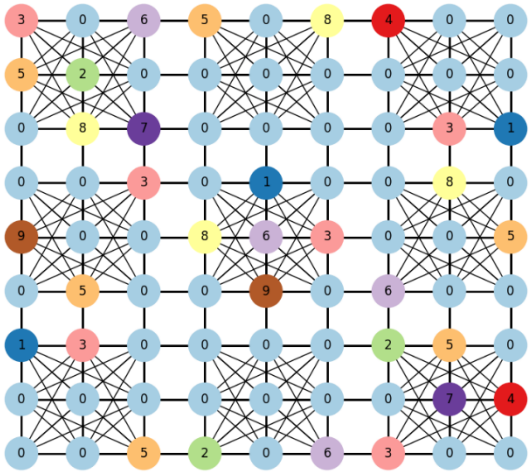
Sumber : dokumentasi pribadi

Untuk implementasi dalam kode, penulis memilih untuk membaca file txt yang berisi sederetan angka 1-9, dan 0. Angka 1-9 menandakan angka yang terisi pada kotak tersebut sementara 0 menandakan kotak kosong. Sebagai contoh file puzzle.txt di bawah ini.

```
3,0,6,5,0,8,4,0,0
5,2,0,0,0,0,0,0,0
0,8,7,0,0,0,0,3,1
0,0,3,0,1,0,0,8,0
9,0,0,8,6,3,0,0,5
0,5,0,0,9,0,6,0,0
1,3,0,0,0,0,2,5,0
0,0,0,0,0,0,0,7,4
```

```
0,0,5,2,0,6,3,0,0
```

Akan diubah menjadi graf sebagai berikut



Gambar 3.5 Pemodelan Sudoku menjadi sebuah graf

Sumber : dokumentasi pribadi

Berikut adalah cuplikan kode untuk pemodelan sudoku pada graf dengan Bahasa python.

```
import numpy as np
import matplotlib as mpl
import networkx as nx
import matplotlib.pyplot as plt


def build_graph(puzzle):
    n = 3
    G = nx.sudoku_graph(n)
    mapping = dict(zip(G.nodes(), puzzle.flatten()))
    pos = dict(zip(list(G.nodes()), nx.grid_2d_graph(n * n, n * n)))
    pos = {node: (pos[node][1], -pos[node][0]) for node in pos}


    # we map the nodes 1-9 to a colormap
    low, *_, high = sorted(mapping.values())
    norm = mpl.colors.Normalize(vmin=low, vmax=high, clip=True)
    mapper = mpl.cm.ScalarMappable(norm=norm, cmap=mpl.cm.Paired)


    # draw the graph
    plt.figure(figsize=(8, 8))
    nx.draw(
        G,
        labels=mapping,
        pos=pos,
        with_labels=True,
        node_color=[mapper.to_rgba(i) for i in mapping.values()],
        width=1,
        node_size=1000,
    )
    plt.show()
    return G
```

B. Menyelesaikan Sudoku dengan Rekursi

Setelah menilik ulang tentang algoritma pewarnaan graf penulis menyadari bahwa algoritma pada pewarnaan graf kurang optimal jika dipakai untuk menyelesaikan permasalahan ini. Hal ini dikarenakan algoritma pewarnaan graf yang ada saat ini masih bersifat greedy sehingga algoritma tidak selalu menghasilkan bilangan kromatik yang paling optimal. Sementara pada kasus ini kita sudah mengetahui bahwa bilangan kromatiknya adalah 9 karena setiap kotak hanya bisa diisi dengan 9 angka. Kita bukan lagi mencari bilangan kromatiknya tetapi mencari cara mewarnai graf sehingga graf yang terbentuk memenuhi aturan dari permainan sudoku.

Untuk itu penulis memilih menggunakan algoritma rekursif untuk menyelesaikan persoalan ini. Algoritma ini sebenarnya bisa digolongkan ke dalam algoritma pewarnaan graf, namun hanya bisa digunakan saat kita sudah mengetahui bilangan kromatik dari sebuah graf itu sendiri. Algoritma ini hanya mengelompokkan simpul-simpul ke golongan warna dari bilangan kromatik.

Berikut adalah cuplikan kode dari algoritma penyelesaian sudoku.

```
def possible(y,x,n):
    global grid
    for i in range(0,9):
        if grid[y][i]==n:
            return False
    for i in range(0,9):
        if grid[i][x]==n:
            return False
    x0 = (x//3)*3
    y0 = (y//3)*3
    for i in range(0,3):
        for j in range(0,3):
            if grid[y0+i][x0+j]==n:
                return False
    return True

def solve():
    global grid
    for y in range(9):
        for x in range(9):
            if grid[y][x]==0:
                for n in range(1,10):
                    if possible(y,x,n):
                        grid[y][x]=n
                        solve()
                        grid[y][x]=0
                return
```

```
        build_graph(grid)
        print(np.matrix(grid))
    input("more solutions ?")
```

C. Fungsi-Fungsi Tambahan

Dalam mengimplementasikan sudoku solver ini penulis juga membuat beberapa fungsi tambahan. Yaitu fungsi load_config untuk membaca file txt dan mengubahnya menjadi sebuah matrix.

```
  def load_config(file_path):
      global grid
      grid = []
      with open(file_path, 'r') as file:
          for line in file:
              row     =     list(map(int,
line.strip().split(',')))
              grid.append(row)
      grid = np.asarray(grid)
      print(grid)
    return grid
```

Dan fungsi main untuk menjalankan fungsi fungsi yang sudah dibuat.

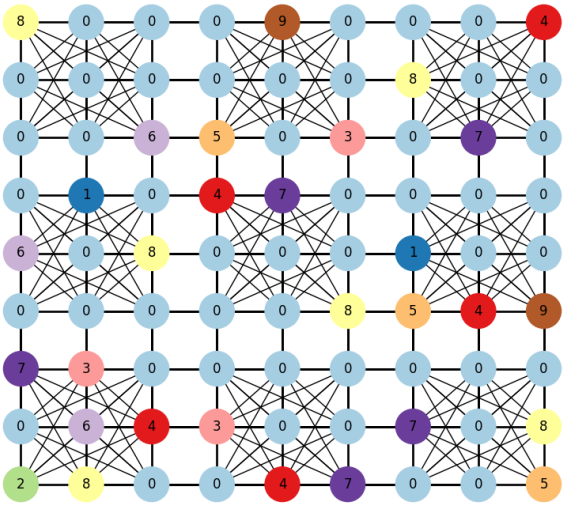
```
  def main():
      file_path = 'tes.txt'
      try:
          global grid
          grid = load_config(file_path)
          test = np.asarray(test)
          build_graph(grid)
          solve()
      except FileNotFoundError:
          print(f"File     not     found:
{file_path}")
      except Exception as e:
          print(f"An    error    occurred:
{str(e)}")
  if __name__ == "__main__":
    main()
```

D. Hasil Tes

Berikut adalah hasil tes penjalanan program pada file puzzle.txt yang berisi state awal sudoku yang diambil dari sudoku generator pada internet.

```
8,0,0,0,9,0,0,0,4
0,0,0,0,0,0,8,0,0
0,0,6,5,0,3,0,7,0
0,1,0,4,7,0,0,0,0
6,0,8,0,0,0,1,0,0
0,0,0,0,0,8,5,4,9
7,3,0,0,0,0,0,0,0
0,6,4,3,0,0,7,0,8
2,8,0,0,4,7,0,0,5
```

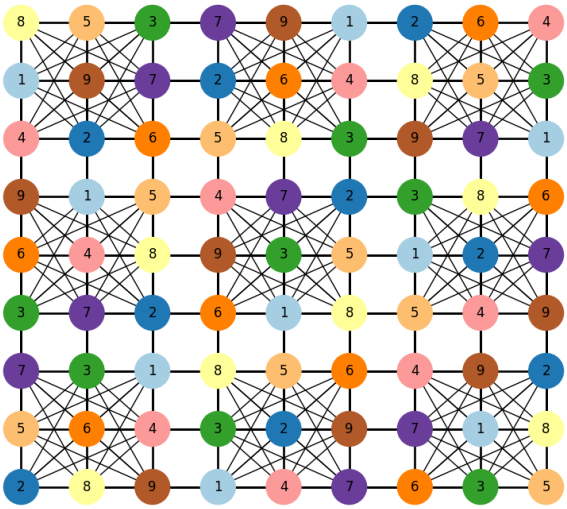
Berikut adalah representasi puzzle.txt dalam graf:



Gambar 3.6 Representasi puzzle.txt pada graf

Sumber : dokumentasi pribadi

Lalu untuk solusi dari sudoku:



Gambar 3.7 Solusi dari puzzle.txt

Sumber : dokumentasi pribadi

IV. KESIMPULAN

Dari percobaan yang telah dilakukan dapat dilihat bahwa persoalan yang disajikan pada permainan sudoku dapat dimodelkan menjadi sebuah graf. Lalu, untuk mendapatkan solusinya kita perlu melakukan pewarnaan kepada graf.

Untuk algoritma penyelesaiannya sendiri, karena kebanyakan algoritma pewarnaan graf bersifat greedy,maka tidak cocok jika digunakan untuk menyelesaikan masalah seperti sudoku yang memerlukan backtracking dan tidak selesai dalam sekali jalan. Oleh karena itu penulis memilih untuk menggunakan algoritma rekursif yang dibuat khusus untuk menyelesaikan permasalahan ini.

V. UCAPAN TERIMA KASIH

Dengan penuh rasa syukur, penulis ingin mengucapkan puji dan syukur kepada Tuhan Yang Maha Esa atas segala rahmat, karunia, serta taufik dan hidayah-Nya, yang telah memandu dan

memberikan keberkahan sehingga penulis berhasil menyelesaikan makalah berjudul "Penerapan Teori Graf dan Rekursi dalam Penyelesaian Permainan Sudoku ". Makalah ini disusun sebagai bagian dari tugas mata kuliah Matematika Diskrit IF2120.

Penulis juga mengucapkan terima kasih yang setinggi-tingginya kepada Bapak Dr. Ir. Rinaldi, M.T. dan Bapak Monterico Adrian, S.T., M.T., sebagai dosen pengajar dalam mata kuliah Matematika Diskrit IF2120 Kelas K3. Terima kasih atas dedikasi, bimbingan, dan pengajaran yang telah diberikan selama satu semester ini, memberikan kontribusi besar dalam pembentukan pemahaman kami sebagai mahasiswa.Penulis juga ingin menyampaikan permohonan maaf apabila terdapat kesalahan dalam penulisan makalah ini. Semoga makalah ini dapat bermanfaat dan memberikan sumbangan positif dalam pemahaman mata kuliah Matematika Diskrit IF2120.

DAFTAR PUSTAKA

[1] Rinaldi Munir, "Graf Bagian 1", https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/19-Graf-Bagian1-2023.pdf (diakses pada tanggal 11 Desember 2023).

[2] Rinaldi Munir, "Graf Bagian 2", https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/20-Graf-Bagian2-2023.pdf (diakses pada tanggal 11 Desember 2023).

[3] Rinaldi Munir. "Graf Bagian 3", https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/21-Graf-Bagian3-2023.pdf (diakses pada tanggal 11 Desember 2023).

[4] Alejandra Ortiz Duran. "Solving Sudoku puzzles with Graph Theory", https://community.wolfram.com/groups/-/m/t/2983903 (diakses pada tanggal 11 Desember 2023).

[5] Anisa Medina Sari, "Algoritma Greedy: Pengertian ,Jenis dan Contoh Program", https://fikti.umsu.ac.id/algoritma-greedy-pengertian-jenis-dan-contoh-program/ (diakses pada tanggal 11 Desember 2023).

[6] NetworkX, "Sudoku and Graph Coloring", https://networkx.org/nx-guides/content/generators/sudoku.html (diakses pada tanggal 11 Desember 2023).

PERNYATAAN

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

Bandung, 11 Desember 2023

Samy Muhammad Haikal, 13522151