Penerapan Teori Himpunan untuk Memecahkan Teka-Teki Masyu

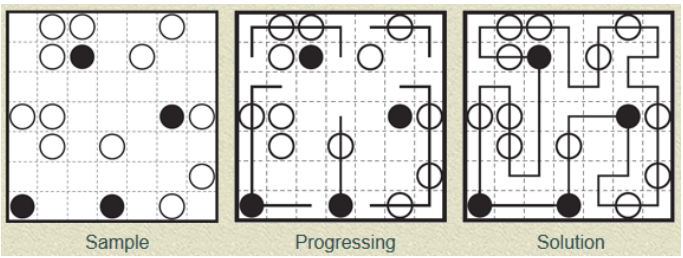
Jonathan Yudi Gunawan 13518084
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13518084@std.stei.itb.ac.id

***Abstrak*—Masyu adalah sebuah teka-teki logika yang pertama kali diterbitkan pada tahun 2000 oleh Nikoli. Tidak seperti teka-teki pada umumnya, Masyu tidak menggunakan angka sebagai petunjuknya, namun menggunakan dua jenis lingkaran sederhana berwarna hitam dan putih yang melambangkan mutiara. Aturan pada Masyu cukup sederhana, namun strategi yang digunakan cukup kompleks. Makalah ini berisi pembahasan mengenai strategi pemecahan teka-teki Masyu dengan memanfaatkan teori himpunan yaitu inklusi dan eksklusi.**

Keywords*—Inklusi dan Ekslusi, Masyu, *Puzzle*, *Solver

I. PENDAHULUAN

Masyu adalah sebuah teka-teki berbentuk matriks berukuran r x c (mulai dari 6x6 hingga 20x20, bahkan lebih) yang berisi dua jenis petunjuk yaitu lingkaran hitam dan lingkaran putih. Tujuan dari teka-teki ini adalah membentuk sebuah gelang atau sirkuit sederhana yang melalui pusat petak secara horisontal atau vertikal. Sebuah gelang sederhana tidak boleh saling bersilangan, membentuk cabang, atau melalui petak yang sama dua kali. Gelang yang dibentuk harus melalui seluruh mutiara yang ada di papan dengan aturan tertentu.[1]



Gambar 1 Contoh Teka Teki Masyu dan Penyelesaiannya
Sumber: http://www.nikoli.co.jp/en/puzzles/masyu.html
(diakses pada 30 November 2019)

Masyu pertama kali diciptakan oleh 矢野龍王 (Yano Ryuoh) and アセトニトリル (“Acetonitrile”) dengan nama *Shiroshinju Kuroshinju* (白真珠黒真珠, yang berarti "mutiara putih dan mutiara hitam"), namun namanya berubah menjadi Masyu karena adanya kesalahan pembacaan oleh presiden Nikoli.[2]

II. TEORI DASAR

A. *Himpunan*

Himpunan (set) adalah kumpulan objek-objek yang berbeda. Objek di dalam himpunan disebut sebagai elemen, unsur, atau anggota. (Munir, 2003). Himpunan tidak memperhatikan urutan elemen-elemennya. Penyajian himpunan dapat dilakukan dengan empat cara, yaitu:

1. Enumerasi

Enumerasi mendata setiap anggota himpunan secara rinci. Jika anggota himpunan terlalu banyak namun mengikuti suatu pola tertentu, dapat digunakan tanda elipsis (...), contohnya :

A = {1,2,3,4}
B = {-2, -1, 0, 1, … , 99, 100}

Himpunan juga memungkinkan terdapat himpunan lagi di dalamnya, seperti:

C = {a, b, {a}, {a, b}, {a, {a}}}

Hal ini menimbulkan konsep baru juga mengenai keanggotaan, misalnya:

Bila
H1 = {a, b}
H2 = {{a, b}}
H3 = {{{a, b}}, {}}

maka a ∈ H1, a ∉ H2, H1 ∈ H2, H1 ∉ H3, H2 ∈ H3, {} ∈ H3.

2. Simbol-simbol Baku

P = himpunan bilangan bulat positif = { 1, 2, 3, ... }
N = himpunan bilangan alami (natural) = { 1, 2, ... }
Z = himpunan bilangan bulat = { ..., -2, -1, 0, 1, 2, ... }
Q = himpunan bilangan rasional
R = himpunan bilangan riil
C = himpunan bilangan kompleks

Himpunan yang universal: semesta, disimbolkan dengan U. Contoh: Misalkan U = {1, 2, 3, 4, 5} dan A adalah himpunan bagian dari U, dengan A = {1, 3, 5}.

3. Notasi Pembentuk Himpunan

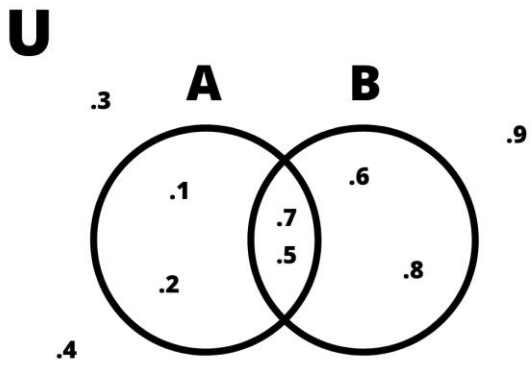
Himpunan dibuat dengan cara mendeskripsikan sifat-sifat yang harus dipenuhi oleh setiap anggota himpunan tersebut. contohnya :

E = {x | x adalah bilangan genap}
A = {y | y ∈ P, y < 10} atau
A = {y | y adalah bilangan bulat positif kurang dari 10}
X = {v | v adalah mahasiswa yang pernah menonton anime Sword Art Online}

4. Diagram Venn

Anggota himpunan disajikan dengan pemodelan dengan diagram berbentuk lingkaran, contohnya:

Misalkan
U = {0,1,2,3,4,5,6,7,8,9},
A = {1,2,5,7}, dan
B = {5,6,7,8},
maka diagram venn yang merepresentasikan himpunan U, A, dan B adalah:



Gambar 2 Diagram venn dari contoh di atas

Ada beberapa prinsip yang digunakan dalam himpunan, di antaranya:

1. Kardinalitas
   Kardinalitas adalah jumlah elemen di dalam suatu himpunan. Notasi yang digunakan adalah:
   n(A) atau |A|
   dengan A adalah nama himpunan tersebut

   misalnya:
   A = {x | x adalah bilangan prima yang lebih kecil dari 20}
   atau
   A = {2,3,5,7,11,13,17,19},
   maka n(A) = |A| = 8

   B = {a, Jojo, 99, Kucing, 13518084}
   maka
   n(B) = |B| = 5

   C = {}
   N(C) = |C| = 0

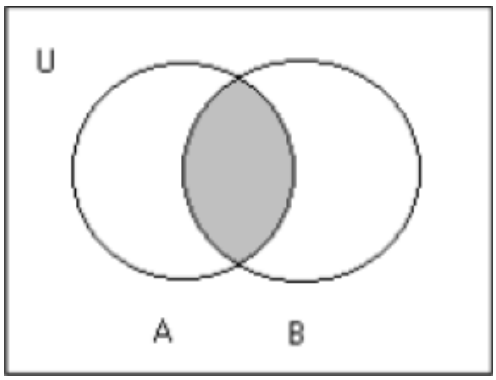
2. Himpunan Kosong (*null set*)
   Himpunan dengan kardinalitas = 0 disebut himpunan kosong (*null set*). Notasi yang digunakan adalah {} atau ∅

3. Himpunan Bagian (*subset*)
   Notasi yang digunakan adalah:
   A ⊆ B
   Himpunan A dikatakan himpunan bagian dari himpunan B jika dan hanya jika setiap elemen A merupakan elemen dari B. Secara formal:
   A ⊆ B ∀x (x ∈ A → x ∈ B)

   Bila A adalah *subset* dari B, maka B adalah *superset* dari A

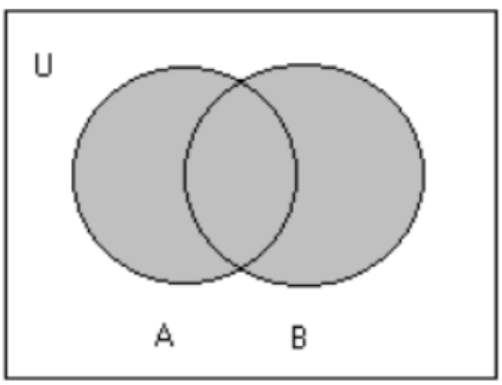
B. Operasi Himpunan

1. Irisan (*Intersection*)
   Operasi irisan A ∩ B setara dengan A dan B. Irisan merupakan himpunan baru yang anggotanya terdiri dari anggota yang dimiliki bersama antara dua atau lebih himpunan yang terhubung. Jika A ∩ B = ∅, maka A dan B dapat dikatakan terpisah (*disjoint*). Contoh:
   {1, 2, 3, 4} ∩ {1, 2} = {1, 2}
   {1} ∩ {1, 2} = {1}
   {Aku} ∩ {Kamu} = {}



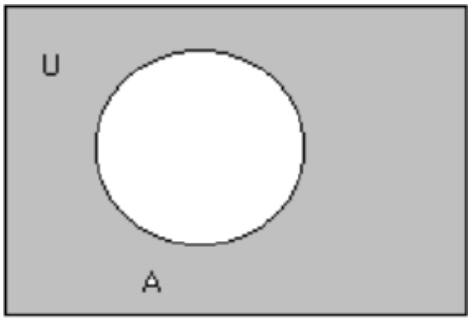
Gambar 3 Diagram Venn Irisan A dengan B [5]

2. Gabungan (*Union*)
   Dua himpunan atau lebih yang digabungkan bersama-sama. Operasi gabungan A ∪ B setara dengan A atau B, dan anggota himpunannya adalah semua anggota yang termasuk himpunan A ataupun B. Contoh:
   {1, 2, 3, 4} ∩ {1, 2} = {1, 2, 3, 4}
   {1} ∩ {1, 2} = {1, 2}
   {Aku} ∩ {Kamu} = {Aku, Kamu}



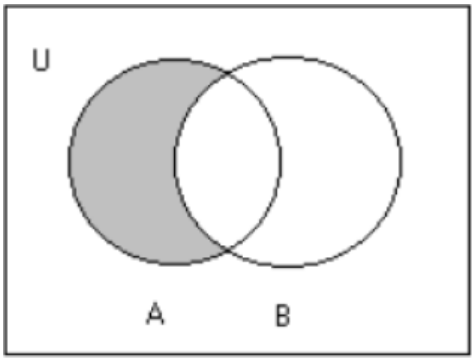
Gambar 4 Diagram Venn Gabungan A dengan B [5]

3. Komplemen (*Complement*)
   Operasi pelengkap A^C / setara dengan bukan A atau A'. Operasi komplemen merupakan operasi yang anggotanya terdiri dari anggota di luar himpunan tersebut, contohnya:
   Misalkan
   U = {1, 2, 3, 4, 5, 6, 7, 8, 9}
   1. Jika A = {1, 2, 3, 4}, maka A^C = {5, 6, 7, 8, 9}
   2. Jika A = {1, 3, 5, 7, 9}, maka A^C = {2, 4, 6, 8}



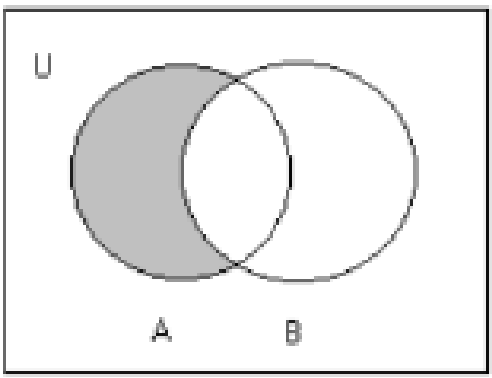
Gambar 5 Diagram Venn Komplemen A [5]

4. Selisih (*Difference*)
   Operasi selisih artinya membentuk sebuah himpunan baru yang berisi anggota A yang tidak dimiliki B.
   Notasi yang digunakan adalah
   $A - B = \{ x \mid x \in A \text{ dan } x \notin B \} = A \cap B^C$
   contohnya :
   Jika A = {1,2,3,4,5} dan B = {1,3,5},
   maka A-B = {2,4} dan B-A = {}



Gambar 6 Diagram Venn Selisih A dengan B [5]

5. Beda Setangkup (*Symmetric Difference*)
   Operasi beda setangkup berarti membentuk sebuah himpunan baru yang berisi anggota A dan anggota B tanpa keduanya.
   Notasi yang digunakan adalah:
   $A \oplus B = (A \cup B) - (A \cap B) = (A - B) \cup (B - A)$
   contohnya:
   Jika A = { 2, 4, 6 } dan B = { 2, 3, 5 },
   maka $A \oplus B$ = { 3, 4, 5, 6 }



Gambar 7 Diagram Venn Beda Setangkup A dengan B [5]

6. Perkalian Kartesian (*Cartesian Product*)
   Operasi ini berarti membentuk sebuah himpunan baru yang elemennya merupakan tuple dari 1 anggota A dan 1 anggota B.
   Notasi yang digunakan adalah:
   $A \times B = \{(a, b) \mid a \in A \text{ dan } b \in B \}$
   contohnya:
   (i) Misalkan C = { 1, 2, 3 }, dan D = { a, b },
   maka C × D = {(1, a), (1, b), (2, a), (2, b), (3, a), (3, b) }
   (ii) Misalkan A = B = himpunan semua bilangan riil,
   maka A × B = himpunan semua titik di bidang datar

Catatan:
1. Jika A dan B merupakan himpunan berhingga,
   maka: $|A \times B| = |A| \, . \, |B|$
2. $(a, b) \neq (b, a)$
3. $A \times B \neq B \times A$ dengan syarat A atau B tidak kosong.
   Pada Contoh (i) di atas, C = { 1, 2, 3 }, dan D = { a, b }
   a. D × C = {(a, 1), (a, 2), (a, 3), (b, 1), (b, 2), (b, 3) }
   b. C × D = { (1, a), (1, b), (2, a), (2, b), (3, a), (3, b) }
   c. $D \times C \neq C \times D$.
4. Jika A = $\{\}$ atau B = $\{\}$, maka $A \times B = B \times A = \{\}$

C. Inklusi dan Eksklusi

Prinsip Inklusi dan Ekslusi merupakan perluasan dari operasi irisan dan gabungan. Misalnya berapa banyak aggota di dalam gabungan dua buah himpunan A dan B? Penggabungan dua himpunan akan menghasilkan himpunan baru dengan jumlah elemen sebanyak A ditambah dengan b, namun mungkin saja himpunan A memiliki elemen-elemen yang sama dengan himpunan B sehingga banyaknya elemen yang baru adalah:
$|A \cup B| = |A| + |B| - |A \cap B|$

Prinsip ini dapat dirampatkan dengan melibatkan tiga buah himpunan, sehingga berlaku persamaan:

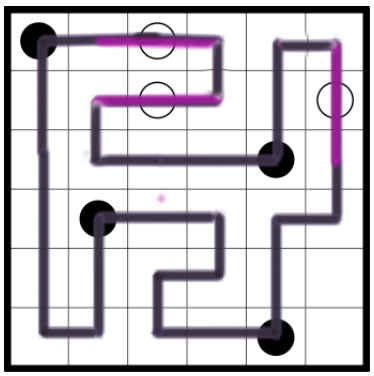
$$\left|A_1 \cup A_2 \cup \ldots \cup A_r\right| = \sum_i \left|A_i\right| - \sum_{1 \le i \le j \le r} \left|A_i \cap A_j\right| + \sum_{1 \le i \le j \le k \le r} \left|A_i \cap A_j \cap A_k\right| + \ldots + (-1)^{r-1} \left|A_1 \cap A_2 \cap \ldots \cap A_r\right|$$

Gambar 8 Persamaan Inklusi Ekslusi pada r buah Himpunan [5]

D. *Aturan Masyu*

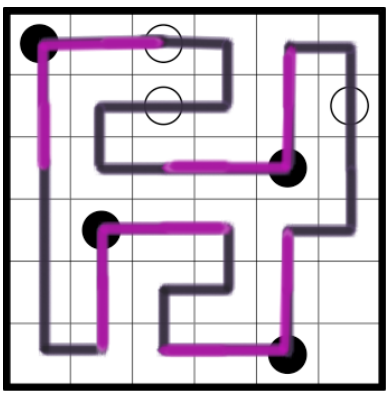
Permainan Masyu melibatkan papan yang dibagi menjadi r x c petak, beberapa di antaranya memiliki petunjuk berupa mutiara berwarna putih atau hitam. Pada papan harus terbentuk sebuah gelang sederhana yang melewati seluruh mutiara serta tidak saling bersilangan. Garis pada gelang hanya boleh berbelok 90 derajat pada pusat petak. Masing-masing mutiara memiliki aturan sebagai berikut:

1. Mutiara putih harus dilewati oleh gelang dengan sebuah garis lurus (vertikal / horisontal) yang dilanjutkan dengan minimal sebuah belokan dengan sudut siku (90 derajat).[1]



Gambar 9 Aturan untuk Mutiara Putih
Sumber: https://krazydad.com/masyu/tutorial/
(diakses pada 1 Desember 2019)

2. Mutiara hitam harus dilewati oleh gelang dengan belokan 90 derajat pada petak tersebut, yang dilanjutkan dengan garis lurus pada kedua sisi nya. [2]



Gambar 10 Aturan untuk Mutiara Hitam
Sumber: https://krazydad.com/masyu/tutorial/
(diakses pada 1 Desember 2019)

E. *Algoritma Pemecahan Masyu*

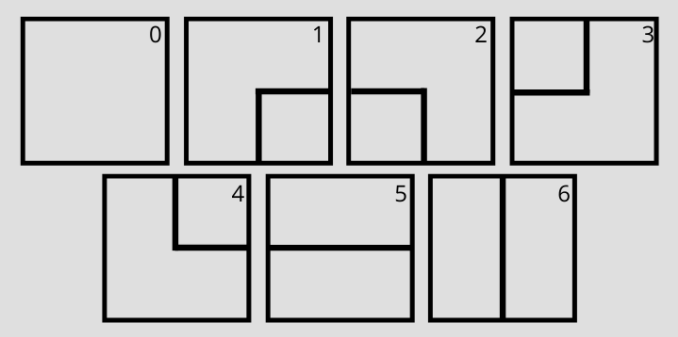
Ada beberapa algoritma yang dapat digunakan untuk memecahkan teka-teki masyu, diantaranya:

1. Algoritma berbasis koneksi
2. Algoritma berbasis batasan

Pada makalah ini akan dibahas mengenai algoritma berbasis batasan. Batasan yang dimaksud adalah batasan bentuk yang mungkin pada masing-masing petak sebagai suatu himpunan solusi yang mungkin.

III. PEMBAHASAN

Pada teka-teki ini, masing-masing petak hanya akan memiliki 7 bentuk garis yang mungkin (lihat gambar 11). Bentuk garis ini yang akan menjadi tolok ukur pada bagian-bagian berikutnya.



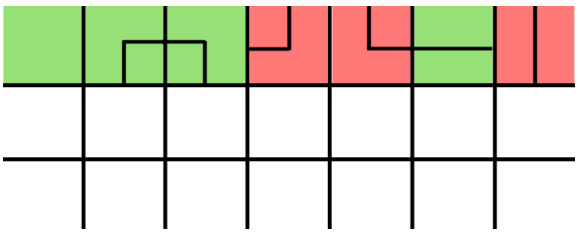
Gambar 11 Batasan Bentuk Garis pada Teka-Teki Masyu

Di awal permainan dapat dilakukan penyaringan atau pembatasan bentuk garis yang mungkin hanya berdasarkan topologi papan dan posisi mutiara putih dan hitam pada papan.

Pembatasan tersebut dapat dilakukan dengan dua cara yaitu irisan (*intersection*) dan selisih (*difference*). Selain itu juga akan dilakukan penggabungan (*union*) pada beberapa kasus untuk menggabungkan beberapa aturan pada sebuah petak.

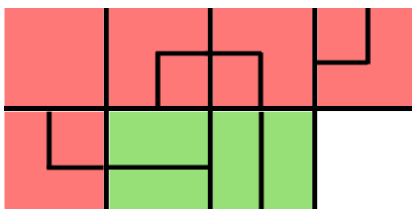
Aturan tersebut antara lain:

1. Pembatasan pada sisi papan.
   Pada sisi papan hanya akan terdapat bentuk yang mengarah ke dalam papan, atau minimal mengarah ke sisi papan pada petak yang lain. Sebagai contoh pada sisi papan sebelah atas, solusi tidak mungkin memuat bentuk 6, bentuk 3, maupun bentuk 4.



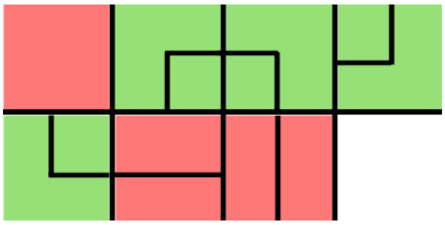
Gambar 12 Visualisasi Aturan 1

2. Pembatasan pada mutiara putih
   Pada mutiara putih, solusi tidak mungkin memuat garis yang bengkok, karena akan melanggar aturan dari permainan itu sendiri, sehingga bentuk 1 hingga 4 bukanlah bagian dari himpunan solusi pada petak tersebut. Selain itu bentuk 0 (petak kosong) juga bukan merupakan himpunan dari solusi, karena petak yang berisi mutiara harus dilewati oleh gelang yang dibuat.



Gambar 13 Visualisasi Aturan 2

3. Pembatasan pada mutiara hitam
   Pada mutiara hitam, solusi tidak mungkin memuat garis lurus, sehingga bentuk 5 dan 6 bukanlah bagian dari himpunan solusi dari petak tersebut. Dengan alasan yang sama seperti mutiara putih, bentuk 0 juga bukan merupakan himpunan dari solusi.



Gambar 14 Visualisasi Aturan 3

Penulis mencoba mengimplementasikan algoritma berbasis batasan ini dalam bahasa Python dengan memanfaatkan tipe data *set* yang tersedia.

```
all_dir        = {0, 1, 2, 3, 4, 5, 6}
left_dir       = {2, 3, 5}
right_dir      = {1, 4, 5}
up_dir         = {3, 4, 6}
down_dir       = {1, 2, 6}
straight_dir   = {5, 6}
bend_dir       = {1, 2, 3, 4}
horizontal_dir = {5}
vertical_dir   = {6}
```

Gambar 15 Implementasi Himpunan Solusi yang Mungkin untuk Masing-masing Petak pada Bahasa Python

Seluruh himpunan solusi yang mungkin kemudian disimpan dalam sebuah array r x c. Misal himpunan solusi pada petak (1,2) disimpan dalam `solution[1][2]`.

```
self.solution = [[all_dir.copy() for x in range(self.r)]
                                 for y in range(self.c)]
```

```
{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|
{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|
{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|
{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|
{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|{ 0,1,2,3,4,5,6}|
```

Gambar 16 Implementasi dan Visualisasi Penyimpanan Himpunan Solusi untuk Papan berukuran 5x5 dalam Program Komputer

Pada Python, operasi pada himpunan dapat dilakukan dengan menggunakan operator primitif, di antaranya:

1. & untuk operasi irisan (*intersection*)
2. - untuk operasi selisih (*difference*)
3. | untuk operasi gabungan (*union*)

Untuk menerapkan aturan pertama, digunakan operasi selisih. Operasi selisih berfungsi seperti mengeliminasi elemen dari suatu himpunan. Misal untuk teka-teki masyu, digunakan untuk mengeliminasi bentuk yang tidak mungkin pasda suatu petak. Aturan pertama terdiri dari empat tahapan:

1. Pada sisi bagian atas, tidak boleh ada bentuk yang salah satu garisnya mengarah ke atas (*up_dir*).
2. Pada sisi bagian kiri, tidak boleh ada bentuk yang salah satu garisnya mengarah ke kiri (*left_dir*).
3. Pada sisi bagian bawah, tidak boleh ada bentuk yang salah satu garisnya mengarah ke bawah (*down_dir*).
4. Pada sisi bagian kanan, tidak boleh ada bentuk yang salah satu garisnya mengarah ke kiri (*right_dir*).

Untuk menerapkan aturan mutiara, digunakan operasi irisan. Operasi irisan berfungsi untuk membatasi kemungkinan bentuk di suatu petak.

Pada kasus mutiara hitam dilakukan pembatasan hanya bentuk bengkok yang diperbolehkan, serta untuk mutiara hitam yang posisinya berada dekat sisi papan (0 – 1 petak dari sisi papan), berlaku pula pembatasan bahwa garis tidak boleh mengarah ke sisi papan karena akan menabrak papan tersebut dan tidak dapat membentuk suatu gelang.

Pada kasus mutiara putih, dilakukan pembatasan hanya bentuk lurus yang diperbolehkan.

```
# remove invalid edge shape
for c in range(self.c):
    self.solution[0][c]  -= up_dir     # top row
    self.solution[-1][c] -= down_dir   # down row

for r in range(self.r):
    self.solution[r][0]  -= left_dir   # leftmost col
    self.solution[r][-1] -= right_dir  # rightmost col
```

```
# remove straight line from black
if pearl[0] == 0:
    # pearl location: (pearl[1], pearl[2])
    self.solution[pearl[1]][pearl[2]] &= bend_dir
    # remove black near edge
    if pearl[1] + 2 > self.r - 1:
        self.solution[pearl[1]][pearl[2]] &= up_dir
    if pearl[1] - 2 < 0:
        self.solution[pearl[1]][pearl[2]] &= down_dir
    if pearl[2] + 2 > self.c - 1:
        self.solution[pearl[1]][pearl[2]] &= left_dir
    if pearl[2] - 2 < 0:
        self.solution[pearl[1]][pearl[2]] &= right_dir
```

```
# remove bend line from white
elif pearl[0] == 1:
    # pearl location: (pearl[1], pearl[2])
    self.solution[pearl[1]][pearl[2]] &= straight_dir
```

Gambar 16 Implementasi Ketiga Aturan di atas dalam Bahasa Python

Setelah menerapkan ketiga aturan di atas, dapat diterapkan aturan mutiara yang implisit. Salah satu aturan tersebut adalah di tengah siku mutiara hitam, hanya boleh memiliki bentuk sama seperti pada mutiara hitam tersebut atau bentuk 0 (kosong).

Sedangkan untuk mutiara putih, bila salah satu petak tidak memiliki bentuk berbelok, maka petak di seberangnya harus memiliki bentuk berbelok.

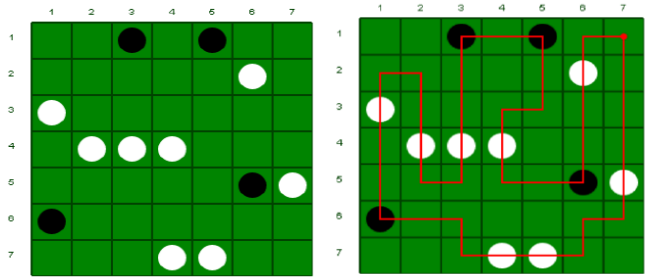
Terakhir, dapat dilakukan ekspansi garis untuk setiap petak yang berisi garis yang belum tersambung. Salah satu cara yang dapat dilakukan adalah dengan menelusuri petak yang sudah memiliki solusi tunggal, lalu melakukan penyaringan himpunan solusi yang mungkin untuk petak-petak di sekitarnya.

Selain cara tersebut, dapat pula digunakan algoritma penelusuran yang lain seperti *Breadth First Search* (*BFS*) atau *Depth First Search* (*DFS*), lalu apabila branch yang dipilih tidak dapat menghasilkan solusi, maka elemen tersebut dihapus dari himpunan solusi pada petak tersebut.

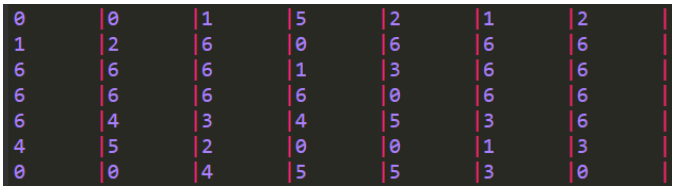
Gambar 17 Implementasi Penyaringan Himpunan Solusi di Sekitar Petak yang Memiliki Solusi Tunggal

Proses penelusuran dan eliminasi himpunan solusi dilakukan secara iteratif dan dilakukan hingga solusi ditemukan.

Ketika himpunan solusi tersebut memiliki kardinalitas satu (1), maka solusi sudah ditemukan untuk petak tersebut. Bila seluruh petak memiliki himpunan solusi berkardinalitas satu (1), maka solusi untuk teka-teki tersebut sudah ditemukan.



Gambar 18 Contoh Teka-teki Masyu pada Papan Berukuran 7x7 beserta Penyelesaiannya



Gambar 19 Pemecahan Teka-teki Masyu pada Gambar 18

Algoritma ini masih belum dapat menyelesaikan seluruh kemungkinan teka-teka Masyu, terlebih lagi bila kerapatan mutiara rendah, karena ada banyak kemungkinan solusi. Namun seharusnya teka-teki yang bagus adalah teka-teki yang solusinya unik / tunggal.

Untuk menyelesaikan teka-teki dengan densitas yang rendah, diperlukan algoritma berbasis koneksi. Namun kekurangan dari algoritma ini adalah algoritma ini tidak semangkus algoritma berbasis batasan, karena algoritma berbasis koneksi mengandalkan prinsip rekursif dengan runut-balik (*backtracking*).

V. KESIMPULAN

Teori himpunan memiliki banyak sekali kegunaan yang dapat diaplikasikan dalam kehidupan manusia sehari-hari. Salah satu manfaatnya adalah untuk memecahkan permasalahan atau teka-teki logika seperti Masyu.

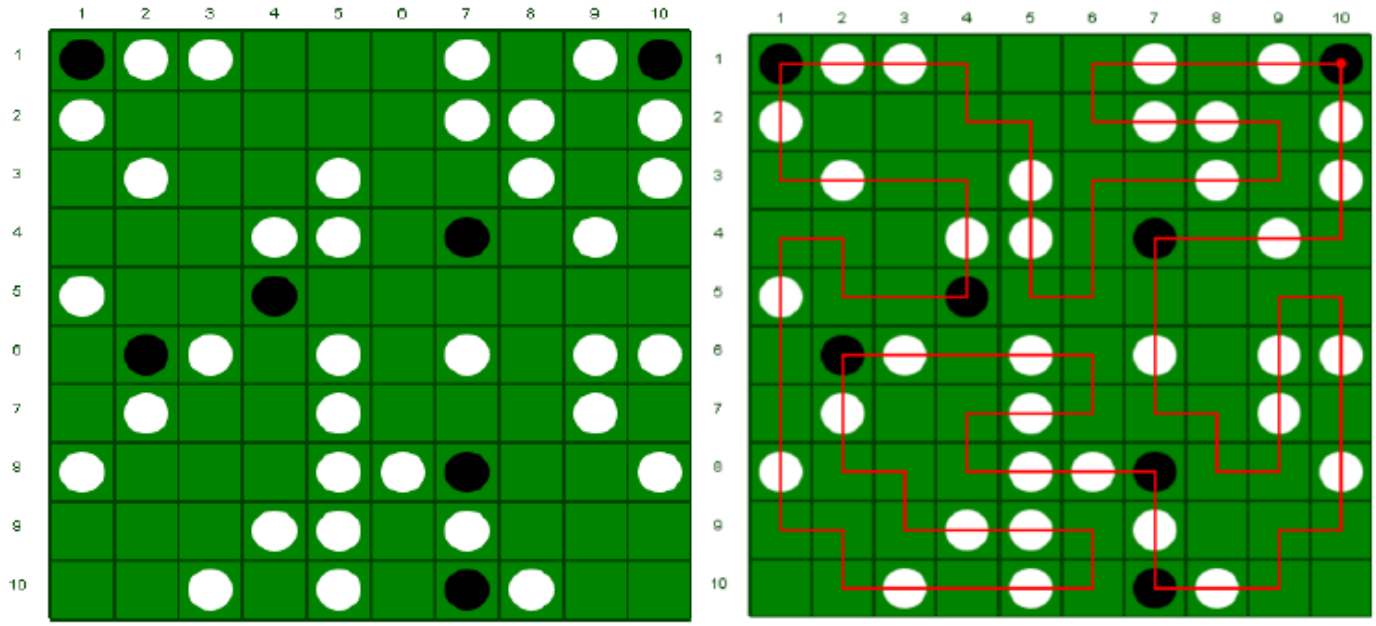
Untuk kerapatan / densitas mutiara yang tinggi, algoritma berbasis batasan akan lebih mangkus daripada algoritma berbasis koneksi.

Algoritma ini dapat dikembangkan lebih lagi untuk dapat diterapkan di bidang lain, misalnya untuk pencarian jalan dengan suatu batasan tertentu, dll.

VI. APENDIKS

Implementasi (*source code*) lengkap dapat dilihat di pranala https://github.com/JonathanGun/masyu-solver.

Berikut adalah contoh hasil eksekusi program pada papan yang lebih besar.



Gambar 20 Contoh Teka-teki Masyu pada Papan Berukuran 10x10

```
Here is the solution:
011---1-10
1-----11-1
-1--1--1-1
---11-0-1-
1--0------
-01-1-1-11
-1--1---1-
1---110--1
---11-1---
--1-1-01--
1       |5       |5       |2       |0       |1       |5       |5       |5       |2
6       |0       |0       |4       |2       |4       |5       |5       |2       |6
4       |5       |5       |2       |6       |1       |5       |5       |3       |6
1       |2       |0       |6       |6       |6       |1       |5       |5       |3
6       |4       |5       |3       |4       |3       |6       |0       |1       |2
6       |1       |5       |5       |5       |2       |6       |0       |6       |6
6       |6       |0       |1       |5       |3       |4       |2       |6       |6
6       |4       |2       |4       |5       |5       |2       |4       |3       |6
4       |2       |4       |5       |5       |2       |6       |0       |1       |3
0       |4       |5       |5       |5       |3       |4       |5       |3       |0

Finished! Time taken: 0.032768249511718750 seconds
```

Gambar 21 Pemecahan Teka-teki Masyu pada Gambar 20

VII. UCAPAN TERIMA KASIH

Puji syukur pada Tuhan yang Maha Esa atas bimbingannya selama pengerjaan makalah. Penulis mengucapkan terima kasih pada kedua orang tua serta teman-teman dari penulis atas segala bentuk dukungan yang telah diberikan. Tidak lupa juga penulis berterima kasih kepada Ibu Fariska selaku dosen pengampu mata kuliah IF2120 Matematika Diskrit kelas 3 dan kepada Bapak Rinaldi Munir atas segala referensi yang telah dipersiapkan bagi penulis.

DAFTAR PUSTAKA

[1] http://www.nikoli.co.jp/en/puzzles/masyu.html diakses pada 30 November 2019

[2] https://www.gmpuzzles.com/blog/masyu-rules-and-info/ diakses pada 30 November 2019

[3] L. John, "Parallelization of a Masyu Puzzle solver", published online, 2015. Diakses pada 29 November 2019

[4] F. Erich, http://www2.stetson.edu/~efriedma/papers/pearl/pearl.html, published online, 2013.

[5] http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2019-2020/Himpunan-(2019).pdf diakses pada 1 Desember 2019

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, 1 Desember 2019

Jonathan Yudi Gunawan / 13518084