

LAPORAN TUGAS KECIL 3
IF2211 STRATEGI ALGORITMA

Penyelesaian Persoalan 15-Puzzle dengan Algoritma
Branch and Bound



Disusun oleh:

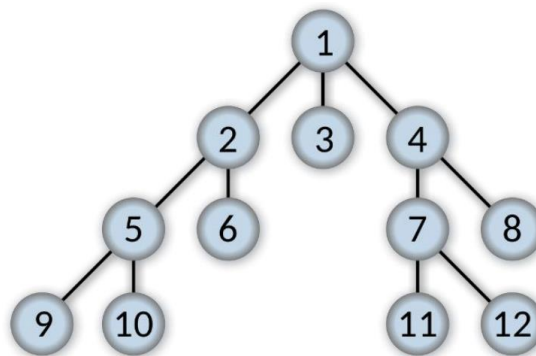
Monica Adelia 13520096

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

2022

1. PENJELASAN ALGORITMA *BRANCH AND BOUND*

Algoritma *Branch and Bound* adalah salah satu algoritma yang digunakan untuk persoalan optimasi. Persoalan optimasi ini tujuannya untuk meminimalkan atau memaksimalkan suatu fungsi objektif yang tidak melanggar batasan (constraints) persoalan. Algoritma *Branch and Bound* ini dapat dibilang merupakan gabungan dari *Breadth First Search* (BFS) dan *least cost search*. Hal ini dikarenakan, *branch and bound* bekerja seperti BFS yaitu simpul yang akan diekspansi berdasarkan urutan pembangkitannya atau bisa disebut FIFO (*First In First Out*). Namun, pada *branch and bound* ini, tiap simpul diberi sebuah nilai *cost*. Nilai *cost* ini merupakan nilai taksiran lintasan termurah ke simpul status tujuan. Simpul berikutnya yang akan diekspansi sudah tidak lagi mengikuti aturan FIFO yaitu berdasarkan urutan pembangkitannya, melainkan simpul yang memiliki *cost* paling kecil untuk kasus minimasi atau *cost* paling besar untuk kasus maksimasi.



Gambar 1 Breadth First Search

Sumber: <https://upload.wikimedia.org/wikipedia/commons/thumb/3/33/Breadth-first-tree.svg/1200px-Breadth-first-tree.svg.png>

Pada algoritma *branch and bound* ini terdapat fungsi pembatas, yaitu “pemangkasan” atau “pembunuhan” node atau jalur yang tidak lagi mengarah pada solusi. Kriteria pemangkasan pada *branch and bound* secara umum adalah nilai simpul tidak lebih baik dari nilai terbaik sejauh ini, simpul tidak merepresentasikan solusi yang *feasible* karena Batasan terlanggar, dan solusi pada simpul tersebut hanya terdiri atas satu titik.

Cara kerja dari algoritma *branch and bound* secara umum ada sebagai berikut, misal terdapat sebuah antrian dan sebuah simpul awal yang akan dilakukan pencarian ke sebuah simpul solusi. Maka,

1. Masukkan simpul akar ke dalam antrian Q. Jika simpul akar adalah simpul solusi (goal simpul), maka solusi telah ditemukan. Stop.
2. Jika Q kosong, tidak ada solusi. Stop.

3. Jika Q tidak kosong, pilih dari antrian Q simpul i yang mempunyai nilai 'cost' $\hat{c}(i)$ paling kecil. Jika terdapat beberapa simpul i yang memenuhi, pilih satu secara sembarang.
4. Jika simpul i adalah simpul solusi, berarti solusi sudah ditemukan, stop. Jika simpul i bukan simpul solusi, maka bangkitkan semua anak-anaknya. Jika i tidak mempunyai anak, kembali ke langkah 2.
5. Untuk setiap anak j dari simpul i, hitung $\hat{c}(j)$, dan masukkan semua anak-anak tersebut ke dalam Q.
6. Kembali ke langkah 2

Implementasi dari perhitungan ongkos dalam penyelesaian 15 puzzle ini adalah sebagai berikut. Costnya merupakan taksiran

$$\hat{c}(P) = f(P) + \hat{g}(P)$$

Dengan $f(P)$ adalah Panjang lintasan simpul akar ke p dan $g(P)$ adalah jumlah ubin tidak kosong yang tidak terdapat pada susunan akhir. Contohnya,

1	2	3	4
5	6		8
9	10	7	11
13	14	15	12

Gambar 2 15 Puzzle

Sumber: <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2019-2020/stima19-20.htm>

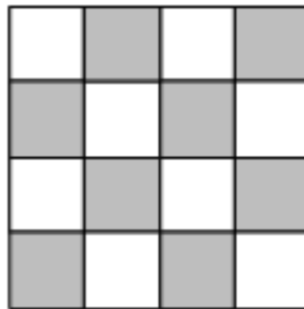
Matrix puzzle diatas, memiliki $g(P)$ sama dengan 3. Angka-angka yang belum berada pada posisi susunan akhir adalah 7, 11, dan 12. Algoritma *branch and bound* menggunakan cost minimum untuk dieksplor terlebih dahulu dalam menyelesaikan permasalahan 15 puzzle.

Matriks puzzle dapat terbagi menjadi dua jenis, dapat diselesaikan dan tidak dapat diselesaikan. Untuk mengetahui apakah sebuah puzzle dapat diselesaikan atau tidak, dapat memanfaatkan fungsi Kurang(i). Fungsi Kurang(i) adalah banyaknya ubin bernomor sedemikian sehingga $j < I$ dan $Posisi(j) > Posisi(i)$. Dalam hal ini, $Posisi(i)$ adalah posisi ubin

bernomor I pada susunan yang sedang diperiksa. Untuk menentukan apakah sebuah puzzle dapat diselesaikan, digunakan rumus

$$\sum_{i=1}^{16} KURANG(i) + X$$

Jadi, semua nilai Kurang(i) mulai dari i=0 sampai i=16 dijumlahkan. Lalu, diakhir dijumlahkan dengan X, dimana X adalah 1 apabila posisi ubin kosong menempati daerah yang diarsir pada gambar dibawah ini dan bernilai 0 apabila ubin terletak di daerah yang tidak diarsir.



Gambar 3 Posisi ubin kosong

Sumber: <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2019-2020/stima19-20.htm>

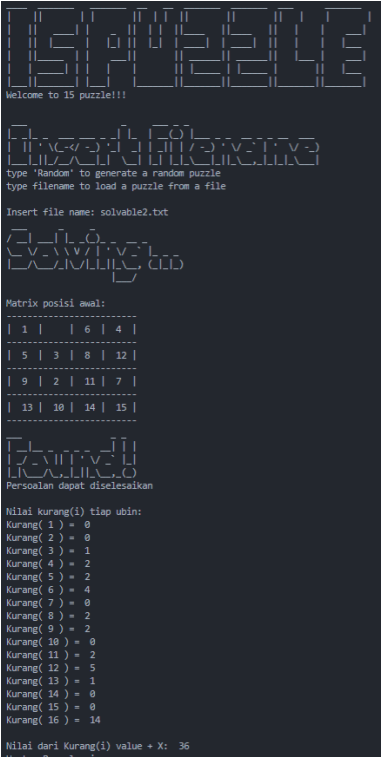
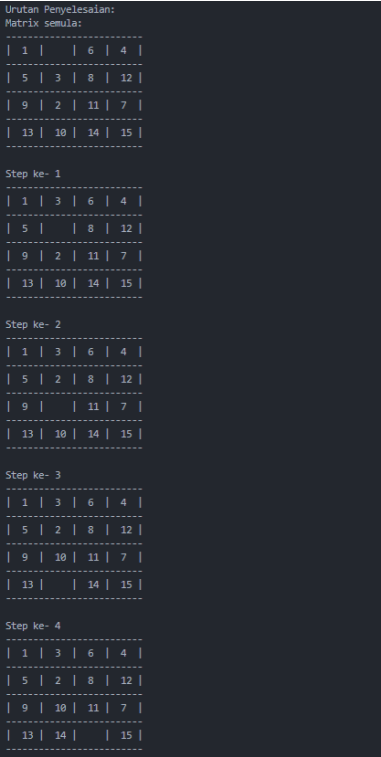
2. SCREEN-SHOT INPUT-OUTPUT PROGRAM

2.1. Tampilan Awal & Input Random/Filename

2.2. Test Case file solvable1.txt

Nama File	<div>solvable1.txt</div> <div> <pre>src > TestCase > solvable1.txt 1 1 2 3 4 2 5 6 16 8 3 9 10 7 11 4 13 14 15 12</pre> </div>
	<div> </div> <div> </div>
Waktu	0.0 detik
Jumlah Node	12
Jumlah Step	3

2.3. Test Case file solvable2.txt

Nama File	<div>solvable2.txt</div> <div>src > TestCase > solvable2.txt</div> <div> <pre> 1 1 16 6 4 2 5 3 8 12 3 9 2 11 7 4 13 10 14 15 </pre> </div>
<div>   </div>	

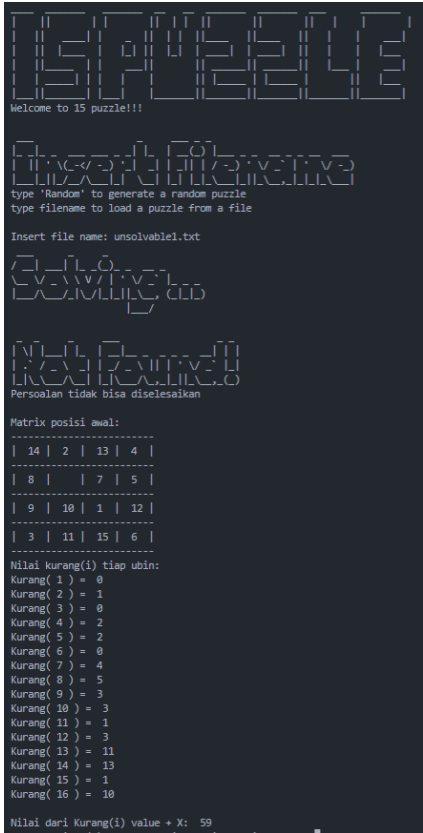
<div> <div> Step ke- 5 <div> <div>1 3 6 4 </div> <div>5 2 8 12 </div> <div>9 10 7 </div> <div>13 14 11 15 </div> </div> </div> <div> Step ke- 6 <div> <div>1 3 6 4 </div> <div>5 2 8 12 </div> <div>9 10 7 </div> <div>13 14 11 15 </div> </div> </div> <div> Step ke- 7 <div> <div>1 3 6 4 </div> <div>5 2 8 </div> <div>9 10 7 12 </div> <div>13 14 11 15 </div> </div> </div> <div> Step ke- 8 <div> <div>1 3 6 4 </div> <div>5 2 8 </div> <div>9 10 7 12 </div> <div>13 14 11 15 </div> </div> </div> <div> Step ke- 9 <div> <div>1 3 4 </div> <div>5 2 6 8 </div> <div>9 10 7 12 </div> <div>13 14 11 15 </div> </div> </div> </div> <div> <div> Step ke- 10 <div> <div>1 3 4 </div> <div>5 2 6 8 </div> <div>9 10 7 12 </div> <div>13 14 11 15 </div> </div> </div> <div> Step ke- 11 <div> <div>1 2 3 4 </div> <div>5 6 8 </div> <div>9 10 7 12 </div> <div>13 14 11 15 </div> </div> </div> <div> Step ke- 12 <div> <div>1 2 3 4 </div> <div>5 6 8 </div> <div>9 10 7 12 </div> <div>13 14 11 15 </div> </div> </div> <div> Step ke- 13 <div> <div>1 2 3 4 </div> <div>5 6 7 8 </div> <div>9 10 12 </div> <div>13 14 11 15 </div> </div> </div> <div> Step ke- 14 <div> <div>1 2 3 4 </div> <div>5 6 7 8 </div> <div>9 10 11 12 </div> <div>13 14 15 </div> </div> </div> </div> <div> <div> Step ke- 15 <div> <div>1 2 3 4 </div> <div>5 6 7 8 </div> <div>9 10 11 12 </div> <div>13 14 15 </div> </div> </div> <div> Waktu eksekusi: 0.009016275405883789 detik Jumlah node yang dibangkitkan: 507 </div> </div>	
Waktu	0.009016275405883789 detik
Jumlah Node	507
Jumlah Step	15

2.4. Test Case file solvable3.txt

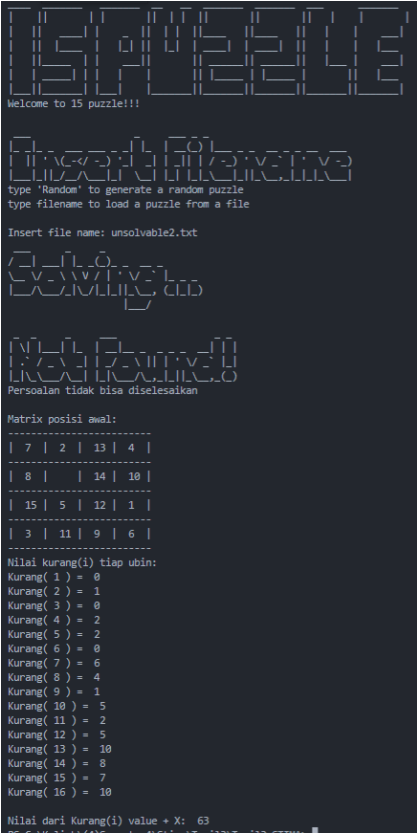
Nama File	<div>solvable3.txt</div> <pre>src > TestCase > solvable3.txt 1 6 2 8 3 2 1 7 4 11 3 5 10 9 12 4 13 14 16 15</pre>
	<div> <pre> Welcome to 15 puzzle!!! type "Random" to generate a random puzzle type filename to load a puzzle from a file Insert file name: solvable3.txt Matrx posisi awal: 6 2 8 3 1 7 4 11 5 10 9 12 13 14 15 Nilai kurang(i) tiap ubin: Kurang(1) = 0 Kurang(2) = 1 Kurang(3) = 1 Kurang(4) = 0 Kurang(5) = 0 Kurang(6) = 5 Kurang(7) = 2 Kurang(8) = 5 Kurang(9) = 0 Kurang(10) = 1 Kurang(11) = 3 Kurang(12) = 0 Kurang(13) = 0 Kurang(14) = 0 Kurang(15) = 0 Kurang(16) = 1 Nilai dari Kurang(i) value + Xi: 20 </pre> </div> <div> <pre> Urutan Penyelesaian: Matrx semula: 6 2 8 3 1 7 4 11 5 10 9 12 13 14 15 Step ke- 1 6 2 8 3 1 7 4 11 5 10 9 12 13 14 15 Step ke- 2 6 2 8 3 1 7 4 11 5 10 9 12 13 14 15 12 Step ke- 3 6 2 8 3 1 7 4 11 5 10 9 11 13 14 15 12 Step ke- 4 6 2 8 3 1 7 14 4 5 10 9 11 13 14 15 12 </pre> </div>

	<div><div>Step ke- 5</div><div><div>6 2 8 3 </div><div>1 7 4 </div><div>5 10 9 11 </div><div>13 14 15 12 </div></div><div>Step ke- 6</div><div><div>6 2 8 3 </div><div>1 10 7 4 </div><div>5 9 11 </div><div>13 14 15 12 </div></div><div>Step ke- 7</div><div><div>6 2 8 3 </div><div>1 10 7 4 </div><div>5 9 11 </div><div>13 14 15 12 </div></div><div>Step ke- 8</div><div><div>6 2 8 3 </div><div>1 10 4 </div><div>5 9 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 9</div><div><div>6 2 3 </div><div>1 10 8 4 </div><div>5 9 7 11 </div><div>13 14 15 12 </div></div></div> <div><div>Step ke- 10</div><div><div>6 2 3 </div><div>1 10 8 4 </div><div>5 9 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 11</div><div><div>6 2 3 </div><div>1 10 8 4 </div><div>5 9 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 12</div><div><div>1 6 2 3 </div><div> 10 8 4 </div><div>5 9 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 13</div><div><div>1 6 2 3 </div><div>5 10 8 4 </div><div> 9 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 14</div><div><div>1 6 2 3 </div><div>5 10 8 4 </div><div>9 7 11 </div><div>13 14 15 12 </div></div></div> <div><div>Step ke- 15</div><div><div>1 6 2 3 </div><div>5 8 4 </div><div>9 10 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 16</div><div><div>1 2 3 </div><div>5 6 8 4 </div><div>9 10 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 17</div><div><div>1 2 3 </div><div>5 6 8 4 </div><div>9 10 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 18</div><div><div>1 2 3 </div><div>5 6 8 4 </div><div>9 10 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 19</div><div><div>1 2 3 4 </div><div>5 6 8 </div><div>9 10 7 11 </div><div>13 14 15 12 </div></div></div> <div><div>Step ke- 20</div><div><div>1 2 3 4 </div><div>5 6 8 </div><div>9 10 7 11 </div><div>13 14 15 12 </div></div><div>Step ke- 21</div><div><div>1 2 3 4 </div><div>5 6 7 8 </div><div>9 10 11 </div><div>13 14 15 12 </div></div><div>Step ke- 22</div><div><div>1 2 3 4 </div><div>5 6 7 8 </div><div>9 10 11 </div><div>13 14 15 12 </div></div><div>Step ke- 23</div><div><div>1 2 3 4 </div><div>5 6 7 8 </div><div>9 10 11 12 </div><div>13 14 15 </div></div><div>Waktu eksekusi: 16.77593207359314 detik Jumlah node yang dibangkitkan: 27184</div></div>
Waktu	16.77593207359314 detik
Jumlah Node	27184
Jumlah Step	23

2.5. Test Case file unsolvable1.txt

Nama File	unsolvable1.txt <pre>src > TestCase > unsolvable1.txt 1 14 2 13 4 2 8 16 7 5 3 9 10 1 12 4 3 11 15 6</pre>
 <pre> Welcome to 15 puzzle!!! type 'Random' to generate a random puzzle type filename to load a puzzle from a file Insert file name: unsolvable1.txt Persoalan tidak bisa diselesaikan Matrix posisi awal: 14 2 13 4 8 16 7 5 9 10 1 12 3 11 15 6 Nilai kurang(i) tiap ubin: kurang(1) = 0 kurang(2) = 1 kurang(3) = 0 kurang(4) = 2 kurang(5) = 2 kurang(6) = 0 kurang(7) = 4 kurang(8) = 5 kurang(9) = 3 kurang(10) = 3 kurang(11) = 1 kurang(12) = 3 kurang(13) = 11 kurang(14) = 13 kurang(15) = 1 kurang(16) = 10 Nilai dari Kurang(i) value + X: 59 </pre>	
Nilai Kurang(i) + X	59

2.6. Test Case file unsolvable2.txt

Nama File	unsolvable2.txt <pre>src > TestCase > unsolvable2.txt 1 7 2 13 4 2 8 16 14 10 3 15 5 12 1 4 3 11 9 6</pre>
	
Nilai Kurang(i) + X	63

2.7. Test Case lain 1 (Random Number Generator)

	
Nilai Kurang(i) + X	41

3. *CHECKLIST*

Poin	Ya	Tidak
1. Program berhasil dikompilasi	√	
2. Program berhasil <i>running</i>	√	
3. Program dapat menerima input dan menuliskan output	√	
4. Luaran sudah benar untuk semua data uji	√	
5. Bonus dibuat		√

4. KODE PROGRAM

4.1. Main.py

```
1  from numpy import *
2
3  from mainFunction import *
4  from helperFunction import *
5  from displayAscii import *
6
7  asciiGameName()
8  puzzlenya = insertFile()
9  asciiSolvingPuzzle()
10
11
12  if(KurangI(puzzlenya)):
13      print("Matrix posisi awal: ")
14      printMatrix(puzzlenya)
15      timeNeeded, node, path = solvePuzzle(puzzlenya)
16
17      print("Waktu eksekusi: ", timeNeeded, " detik")
18      print("Jumlah node yang dibangkitkan: ", node)
19  else:
20      asciiNotFound()
21      print("Matrix posisi awal: ")
22      printMatrix(puzzlenya)
23      kurangItiapUbin(puzzlenya)
24      printKurangIValue(puzzlenya)
```

Bagian ini berisi main program dari 15 puzzle. File ini berisi program yang akan memanggil pembacaan file masukan dari pengguna dan akan melakukan penyelesaian dari puzzle.

4.2. mainFunction.py

File ini terdiri dari fungsi-fungsi utama pada 15 puzzle solver. Terdapat function solvePuzzle, function move, function KurangI, dan structure node dari puzzle.

4.2.1. function solvePuzzle

```
1 def solvePuzzle(puzzlenya):
2     q = PriorityQueue()
3     node = 1
4     path = list()
5     path.append(puzzlenya)
6
7     puzzleMatrix = matrixStruct(sumOfNotMatch(puzzlenya), (node * -1), sumOfNotMatch(puzzlenya), puzzlenya, 0, path)
8
9     answer = [[1,2,3,4], [5,6,7,8], [9,10,11,12], [13,14,15,16]]
10    visited = []
11    inQueue = []
12
13    q.put(puzzleMatrix)
14    inQueue.append(puzzleMatrix.matriksPuzzle)
15    start_time = time.time()
16    found = True
17
18    while not q.empty():
19        end_timeNow = time.time()
20        timeNeeded = end_timeNow - start_time
21        if(timeNeeded > 1800):
22            found = False
23            break
24
25        current = q.get()
26
27        if current.matriksPuzzle not in visited:
28            visited.append(current.matriksPuzzle)
29
30            if(current.matriksPuzzle == answer):
31                break
32            else:
33                row, col = findIndexOfElmnt(current.matriksPuzzle, 16)
34
35                if(row > 0):
36                    node = node + 1
37                    newMatrix = moveUp(current.matriksPuzzle, row, col)
38                    cost = current.depth + 1 + sumOfNotMatch(newMatrix)
39                    newPath = current.path.copy()
40                    newPath.append(newMatrix)
41                    newMatrixStruct = matrixStruct(cost, (node * -1), sumOfNotMatch(newMatrix), newMatrix, current.depth + 1, newPath)
42                    if (newMatrixStruct.matriksPuzzle not in visited) and (newMatrixStruct.matriksPuzzle not in inQueue):
43                        q.put(newMatrixStruct)
44                        inQueue.append(newMatrixStruct.matriksPuzzle)
45
46                if(row < 3):
47                    node = node + 1
48                    newMatrix = moveDown(current.matriksPuzzle, row, col)
49                    cost = current.depth + 1 + sumOfNotMatch(newMatrix)
50                    newPath = current.path.copy()
51                    newPath.append(newMatrix)
52                    newMatrixStruct = matrixStruct(cost, (node * -1), sumOfNotMatch(newMatrix), newMatrix, current.depth + 1, newPath)
53                    if (newMatrixStruct.matriksPuzzle not in visited) and (newMatrixStruct.matriksPuzzle not in inQueue):
54                        q.put(newMatrixStruct)
55                        inQueue.append(newMatrixStruct.matriksPuzzle)
56
57                if(col > 0):
58                    node = node + 1
59                    newMatrix = moveLeft(current.matriksPuzzle, row, col)
60                    cost = current.depth + 1 + sumOfNotMatch(newMatrix)
61                    newPath = current.path.copy()
62                    newPath.append(newMatrix)
63                    newMatrixStruct = matrixStruct(cost, (node * -1), sumOfNotMatch(newMatrix), newMatrix, current.depth + 1, newPath)
64                    if (newMatrixStruct.matriksPuzzle not in visited) and (newMatrixStruct.matriksPuzzle not in inQueue):
65                        q.put(newMatrixStruct)
66                        inQueue.append(newMatrixStruct.matriksPuzzle)
67
68                if(col < 3):
69                    node = node + 1
70                    newMatrix = moveRight(current.matriksPuzzle, row, col)
71                    cost = current.depth + 1 + sumOfNotMatch(newMatrix)
72                    newPath = current.path.copy()
73                    newPath.append(newMatrix)
74                    newMatrixStruct = matrixStruct(cost, (node * -1), sumOfNotMatch(newMatrix), newMatrix, current.depth + 1, newPath)
75                    if (newMatrixStruct.matriksPuzzle not in visited) and (newMatrixStruct.matriksPuzzle not in inQueue):
76                        q.put(newMatrixStruct)
77                        inQueue.append(newMatrixStruct.matriksPuzzle)
78
79        end_time = time.time()
80        timeNeeded = end_time - start_time
81
82        if(found == True):
83            asciFound()
84            kurangItiapUbin(puzzlenya)
85            printKurangIValue(puzzlenya)
86            printUrutanMatrix(current.path)
87        else:
88            asciFailedToSolve();
89            kurangItiapUbin(puzzlenya)
90            printKurangIValue(puzzlenya)
91
92    return timeNeeded, node, current.path
```

4.2.2. matrixStruct dan function KurangI

```
1 class matrixStruct(NamedTuple):
2     cost: int
3     nodeNumber: int
4     notInResult: int
5     matriksPuzzle: matrix
6     depth: int
7     path: list
8
9 def KurangI(matrix):
10     value = 0
11     for i in range(0,4):
12         for j in range(0,4):
13             if((matrix[i][j] == 16) and (i % 2 == 0) and (j % 2 == 1)):
14                 value = value + 1
15             elif ((matrix[i][j] == 16) and (i % 2 == 1) and (j % 2 == 0)):
16                 value = value + 1
17             value = value + findLower(matrix, i, j)
18     if(value % 2 == 0):
19         return True
20     else:
21         return False
```

4.2.3. function moveUp, moveDown, moveRight, dan moveLeft

```
1 def moveUp(matrix, i, j):
2     newMatrix = [[0,0,0,0], [0,0,0,0], [0,0,0,0], [0,0,0,0]]
3     for k in range(4):
4         for l in range(4):
5             newMatrix[k][l] = matrix[k][l]
6
7     temp = newMatrix[i-1][j]
8     newMatrix[i-1][j] = newMatrix[i][j]
9     newMatrix[i][j] = temp
10
11     return newMatrix
12
13 def moveDown(matrix, i, j):
14     newMatrix = [[0,0,0,0], [0,0,0,0], [0,0,0,0], [0,0,0,0]]
15     for k in range(4):
16         for l in range(4):
17             newMatrix[k][l] = matrix[k][l]
18
19     temp = newMatrix[i+1][j]
20     newMatrix[i+1][j] = newMatrix[i][j]
21     newMatrix[i][j] = temp
22
23     return newMatrix
```

```
1 def moveRight(matrix, i, j):
2     newMatrix = [[0,0,0,0], [0,0,0,0], [0,0,0,0], [0,0,0,0]]
3     for k in range(4):
4         for l in range(4):
5             newMatrix[k][l] = matrix[k][l]
6
7     temp = newMatrix[i][j+1]
8     newMatrix[i][j+1] = newMatrix[i][j]
9     newMatrix[i][j] = temp
10
11     return newMatrix
12
13 def moveLeft(matrix, i, j):
14     newMatrix = [[0,0,0,0], [0,0,0,0], [0,0,0,0], [0,0,0,0]]
15     for k in range(4):
16         for l in range(4):
17             newMatrix[k][l] = matrix[k][l]
18
19     temp = newMatrix[i][j-1]
20     newMatrix[i][j-1] = newMatrix[i][j]
21     newMatrix[i][j] = temp
22
23     return newMatrix
```


4.3. helperFunction.py

```
1 import random
2 from numpy import *
3
4 from displayAscii import *
5
6 def getIndex(i,j):
7     return ((4 * i) + j)
8
9 def findLower(matrix, row, col):
10     sum = 0
11     for i in range(0,4):
12         for j in range(0,4):
13             if((getIndex(i,j) > getIndex(row, col)) and (matrix[i][j] < matrix[row][col])):
14                 sum = sum + 1
15     return sum
16
17 def findIndexOfElement(matrix, elmnt):
18     for i in range(0,4):
19         for j in range(0,4):
20             if(matrix[i][j] == elmnt):
21                 return i, j
22
23 def readMyFile(filename):
24     with open(filename, 'r') as f:
25         lines = f.readlines()
26         a = [[int(x) for x in line.split()] for line in lines]
27     return a
28
29 def createRandomPuzzle():
30     a = [[0,0,0,0], [0,0,0,0], [0,0,0,0], [0,0,0,0]]
31     my_list = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]
32     random.shuffle(my_list)
33
34     for i in range(0,4):
35         for j in range(0,4):
36             a[i][j] = my_list[getIndex(i,j)]
37
38     return a
39
40 def sumOfNoMatch(matrix):
41     newMatrix = [[1,2,3,4], [5,6,7,8], [9,10,11,12], [13,14,15,16]]
42
43     sum = 0
44     for i in range(0,4):
45         for j in range(0,4):
46             if(matrix[i][j] != 16):
47                 if(matrix[i][j] != newMatrix[i][j]):
48                     sum = sum + 1
49     return sum
50
51 def insertFile():
52     asciiInsertFileName()
53     print("Type 'Random' to generate a random puzzle")
54     print("Type filename to load a puzzle from a file")
55     print()
56     filename = input("Insert file name: ")
57     if(filename == "Random"):
58         return createRandomPuzzle()
59     else:
60         filename = "./src/TestCase/" + filename
61
62     puzzle = readMyFile(filename)
63
64     return puzzle
65
66 def printMatrix(matrix):
67     print("-----")
68     for i in range(0,4):
69         for j in range(0,4):
70             print("[", end=" ")
71             if(matrix[i][j] < 10):
72                 print(matrix[i][j], end=" ")
73             else:
74                 if(matrix[i][j] == 16):
75                     print(end=" ")
76                 else:
77                     print(matrix[i][j], end=" ")
78         print("]")
79     print("-----")
80
81 def kurangItiapUbin(matrix):
82     print("Nilai kurang(i) tiap ubin: ")
83     for i in range(1,17):
84         row, col = findIndexOfElement(matrix,i)
85         print("kurang( ", i, ") = ", findLower(matrix, row, col))
86
87 def KurangValue(matrix):
88     value = 0
89     for i in range(0,4):
90         for j in range(0,4):
91             if((matrix[i][j] == 16) and (i % 2 == 0) and (j % 2 == 1)):
92                 value = value + 1
93             elif ((matrix[i][j] == 16) and (i % 2 == 1) and (j % 2 == 0)):
94                 value = value + 1
95             value = value + findLower(matrix, i, j)
96     return value
97
98 def printKurangValue(matrix):
99     print()
100     print("Nilai dari kurang(i) value + x: ", KurangValue(matrix))
101
102 def printStatusMatrix(path):
103     print("Urutan Penyelesaian: ")
104     stepCounter = 0
105     for matrix in path:
106         if(stepCounter == 0):
107             print("Matrix semula ")
108             printMatrix(matrix)
109             print()
110         else:
111             print("Step ke-", stepCounter)
112             printMatrix(matrix)
113             print()
114     stepCounter = stepCounter + 1
```

4.4. displayAscii.py

```
1 def asciiGameName():
2     print("_____")
3     print("| | | | | | | | | | | | | | | | |")
4     print("| | | | | | | | | | | | | | | |")
5     print("| | | | | | | | | | | | | | | |")
6     print("| | | | | | | | | | | | | | | |")
7     print("| | | | | | | | | | | | | | | |")
8     print("| | | | | | | | | | | | | | | |")
9     print("Welcome to 15 puzzle!!!")
10    print()
11
12 def asciiInsertFileName():
13    print("_____")
14    print("| | | | | | | | | | | | | | | |")
15    print("| | | | | | | | | | | | | | | |")
16    print("| | | | | | | | | | | | | | | |")
17
18 def asciiSolvingPuzzle():
19    print("_____")
20    print("/ | | | | | | | | | | | | | | |")
21    print("\ \ \ \ \ \ \ \ \ \ \ \ \ \ \")
22    print("| | | | | | | | | | | | | | | |")
23    print(" | | | | | | | | | | | | | | |")
24    print()
25
26 def asciiFound():
27    print("_____")
28    print("| | | | | | | | | | | | | | | |")
29    print("| | | | | | | | | | | | | | | |")
30    print("| | | | | | | | | | | | | | | |")
31    print("Persoalan dapat diselesaikan")
32    print()
33
34 def asciiNotFound():
35    print("_____")
36    print("| | | | | | | | | | | | | | | |")
37    print("| | | | | | | | | | | | | | | |")
38    print("| | | | | | | | | | | | | | | |")
39    print("Persoalan tidak bisa diselesaikan")
40    print()
41
42 def asciiFailedToSolve():
43    print("_____")
44    print("/ | | | | | | | | | | | | | | |")
45    print("| ( ) | | | | | | | | | | | | | |")
46    print("\ \ \ \ \ \ \ \ \ \ \ \ \ \ \")
47    print(" | | | | | | | | | | | | | | |")
48    print("You've waited more than 30 minutes!")
49    print("Try again or Try another puzzle")
50    print()
```

5. CONTOH PERSOALAN 15-PUZZLE

Contoh persoalan 15-puzzle dapat dilihat pada link repository github pada folder src > TestCase. Dalam contoh persoalan ini, angka 16 menandakan puzzle yang kosong. Contoh-contohnya adalah:

No	Nama	Puzzle
1	solvable1	1 2 3 4 5 6 16 8 9 10 7 11 13 14 15 12
2	solvable2	1 16 6 4 5 3 8 12 9 2 11 7 13 10 14 15
3	solvable3	6 2 8 3 1 7 4 11 5 10 9 12 13 14 16 15
4	unsolvable1	14 2 13 4 8 16 7 5 9 10 1 12 3 11 15 6
5	unsolvable2	7 2 13 4 8 16 14 10 15 5 12 1 3 11 9 6

6. GITHUB

<https://github.com/adeliaaaa/Tucil3-STIMA.git>