

# **LAPORAN TUGAS KECIL 3**

## **IF2211 STRATEGI ALGORITMA**

Penyelesaian Persoalan 15-Puzzle Dengan Algoritma Branch and Bound



NIM	: 13520085
Nama	: Ubaidillah Ariq Prathama
Kelas	: K01

**PROGRAM STUDI TEKNIK INFORMATIKA  
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG  
2021**

## A. Algoritma Branch and Bound Untuk Menyelesaikan Permasalahan

15-Puzzle adalah teka-teki geser yang memiliki 15 persegi bernomor 1 sampai 15 dalam bingkai dengan tinggi 4 kotak dan lebar 4 kotak, menyisakan satu posisi ubin kosong. Kotak di baris atau kolom yang sama dari posisi terbuka dapat dipindahkan dengan menggesernya secara horizontal atau vertikal. Tujuan akhir dari game ini adalah membuat kotak tersebut teratur kembali setelah beberapa kali memindahkan.

Algoritma Branch and Bound merupakan algoritma yang membagi permasalahan menjadi sub masalah lebih kecil yang mengarah ke solusi dengan pencabangan (branching) dan melakukan pembatasan (bounding) untuk mencapai solusi optimal. Pencabangan (branching) yaitu proses membentuk permasalahan ke dalam bentuk struktur pohon pencarian (search tree). Proses Pencabangan dilakukan untuk membangun semua cabang pohon yang menuju solusi, sedangkan proses pembatasan dilakukan dengan menghitung estimasi nilai (cost) simpul dengan memperhatikan batas. Pada dasarnya Branch and Bound merupakan BFS dengan optimasi.

15-Puzzle dapat diselesaikan menggunakan algoritma Branch and Bound. Langkah-langkah dalam menyelesaikan persoalan ini adalah :

1. Memeriksa apakah puzzle dapat diselesaikan. Hal ini dapat dilakukan dengan menggunakan fungsi Kurang(i) dan Arsir(x,y) dengan definisi :

$Kurang(i)$  = Banyaknya ubin bernomor  $j$  sedemikian sehingga  $j < i$  dan  $Posisi(j) > Posisi(i)$ .  $Posisi(i)$  = posisi ubin bernomor  $i$  pada susunan yang diperiksa.

$Arsir(x,y)$  = Bernilai 0 jika  $x-y$  genap dan bernilai 1 jika  $x-y$  ganjil,  $x$  dan  $y$  di sini merupakan koordinat awal ubin yang kosong.

Jika  $\sum_{i=1}^{16} Kurang(i) + Arsir(x,y)$  ganjil maka puzzle tidak mungkin diselesaikan. Jika genap maka puzzle dapat diselesaikan. Hal ini diimplementasikan dalam fungsi lokasi, kurang, totalKurang.

2. Menghitung fungsi cost dari root. Fungsi cost dapat didefinisikan sebagai banyaknya ubin yang belum berada pada posisi yang benar ditambah kedalaman dari search tree.

Untuk kasus root, kedalamannya adalah 0. Hal ini dapat diimplementasikan dengan looping matriks dan diimplementasikan dalam fungsi cost.

3. Mencari child yang mungkin dari parent. Terdapat 4 kemungkinan arah berpindahnya kotak kosong (atas, kanan, bawah, kiri). Constraintnya adalah setelah berpindah ubin kosong masih harus berada di dalam kotak 4x4 dan tidak boleh kembali ke posisi awal sebelum parentnya (repetisi). Hal ini dapat dicek dengan mudah dengan percabangan dan diimplementasikan dalam fungsi nextNode.
4. Child yang memungkinkan dari parent akan dicari costnya. Child ini akan dikonstruksi menjadi sebuah class Node yang berisi (parent, id, matrix, cost, block\_pos, level). Setelah itu akan dipush ke dalam priority queue yang akan memprioritaskan Node dengan cost terendah. Priority queue ini diimplementasikan menggunakan struktur data bawaan python yaitu heap. Hal ini diimplementasikan dalam file priorityQueue.py dan fungsi nextNode.
5. Jika semua child dari parent sudah dicek dan dimasukkan ke dalam priority queue, pencarian dilanjutkan. Keluarkan (pop) elemen terkecil dari priority queue dan elemen ini akan menjadi parent yang baru. Lakukan terus hingga mendapatkan solusi. Hal ini diimplementasikan dengan while loop pada file main.py

## B. Hasil Pengujian

Pengujian test1.txt

```
Masukkan nama file : test1.txt
Solusi ditemukan
Simpul ke-0
Cost : 4
Level : 0
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | - | 8 |
-----
| 9 | 10 | 7 | 11 |
-----
| 13 | 14 | 15 | 12 |
-----

Simpul ke-4
Cost : 4
Level : 1
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 10 | - | 11 |
-----
| 13 | 14 | 15 | 12 |
-----
```

```
Simpul ke-6
Cost : 4
Level : 2
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 10 | 11 | - |
-----
| 13 | 14 | 15 | 12 |
-----

Simpul ke-10
Cost : 3
Level : 3
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 10 | 11 | 12 |
-----
| 13 | 14 | 15 | - |
-----

Solusi ditemukan
Waktu Eksekusi : 0.14573097229003906 detik
```

## Pengujian test2.txt

```
Masukkan nama file : test2.txt
Solusi ditemukan
Simpul ke-0
Cost : 11
Level : 0
-----
| 1 | 7 | 2 | 4 |
-----
| 9 | 5 | 3 | 8 |
-----
| 13 | - | 11 | 12 |
-----
| 10 | 6 | 14 | 15 |
-----
```

```
Simpul ke-4
Cost : 12
Level : 1
-----
| 1 | 7 | 2 | 4 |
-----
| 9 | 5 | 3 | 8 |
-----
| 13 | 6 | 11 | 12 |
-----
| 10 | - | 14 | 15 |
-----
```

```
Simpul ke-10
Cost : 13
Level : 2
-----
| 1 | 7 | 2 | 4 |
-----
| 9 | 5 | 3 | 8 |
-----
| 13 | 6 | 11 | 12 |
-----
| - | 10 | 14 | 15 |
-----
```

```
Simpul ke-25
Cost : 13
Level : 3
-----
| 1 | 7 | 2 | 4 |
-----
| 9 | 5 | 3 | 8 |
-----
| - | 6 | 11 | 12 |
-----
| 13 | 10 | 14 | 15 |
-----
```

```
Simpul ke-33
Cost : 13
Level : 4
-----
| 1 | 7 | 2 | 4 |
-----
| - | 5 | 3 | 8 |
-----
| 9 | 6 | 11 | 12 |
-----
| 13 | 10 | 14 | 15 |
-----
```

```
Simpul ke-49
Cost : 13
Level : 5
-----
| 1 | 7 | 2 | 4 |
-----
| 5 | - | 3 | 8 |
-----
| 9 | 6 | 11 | 12 |
-----
| 13 | 10 | 14 | 15 |
-----
```

```
Simpul ke-55
Cost : 14
Level : 6
-----
| 1 | - | 2 | 4 |
-----
| 5 | 7 | 3 | 8 |
-----
| 9 | 6 | 11 | 12 |
-----
| 13 | 10 | 14 | 15 |
-----
```

```
Simpul ke-102
Cost : 14
Level : 7
-----
| 1 | 2 | - | 4 |
-----
| 5 | 7 | 3 | 8 |
-----
| 9 | 6 | 11 | 12 |
-----
| 13 | 10 | 14 | 15 |
-----
```

```
Simpul ke-119
Cost : 14
Level : 8
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 7 | - | 8 |
-----
| 9 | 6 | 11 | 12 |
-----
| 13 | 10 | 14 | 15 |
-----
```

```
Simpul ke-125
Cost : 14
Level : 9
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | - | 7 | 8 |
-----
| 9 | 6 | 11 | 12 |
-----
| 13 | 10 | 14 | 15 |
-----
```

```
Simpul ke-127
Cost : 14
Level : 10
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | - | 11 | 12 |
-----
| 13 | 10 | 14 | 15 |
-----
```

```
Simpul ke-130
Cost : 14
Level : 11
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 10 | 11 | 12 |
-----
| 13 | - | 14 | 15 |
-----
```

```
Simpul ke-132
Cost : 14
Level : 12
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 10 | 11 | 12 |
-----
| 13 | 14 | - | 15 |
-----
```

```
Simpul ke-135
Cost : 13
Level : 13
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 10 | 11 | 12 |
-----
| 13 | 14 | 15 | - |
-----
```

```
Solusi ditemukan
Waktu Eksekusi : 0.5509054660797119 detik
```

## Pengujian test3.txt

```
Masukkan nama file : test3.txt
Solusi ditemukan
Simpul ke-0
Cost : 7
Level : 0
```

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

```
Simpul ke-3
Cost : 8
Level : 1
```

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

```
Simpul ke-6
Cost : 9
Level : 2
```

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

```
Simpul ke-13
Cost : 11
Level : 3
```

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

```
Simpul ke-36
Cost : 11
Level : 4
```

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

```
Simpul ke-50
Cost : 13
Level : 5
```

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

```
Simpul ke-123
Cost : 14
Level : 6
```

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

```
Simpul ke-215
Cost : 15
Level : 7
```

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

```
Simpul ke-407
Cost : 16
Level : 8
```

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

```
Simpul ke-700
Cost : 17
Level : 9
```

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

```
Simpul ke-1334
Cost : 17
Level : 10
```

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

```
Simpul ke-1578
Cost : 17
Level : 11
```

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

```
Simpul ke-1627
Cost : 18
Level : 12
```

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

```
Simpul ke-2898
Cost : 19
Level : 13
```

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

```
Simpul ke-5589
Cost : 19
Level : 14
```

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

```
Simpul ke-5716
Cost : 21
Level : 15
```

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

```

Simpul ke-19368
Cost : 21
Level : 16
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 14 | - | 12 |
-----
| 13 | 11 | 10 | 15 |
-----

Simpul ke-19854
Cost : 22
Level : 17
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 14 | 10 | 12 |
-----
| 13 | 11 | - | 15 |
-----

Simpul ke-37444
Cost : 23
Level : 18
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 14 | 10 | 12 |
-----
| 13 | - | 11 | 15 |
-----

Simpul ke-72071
Cost : 23
Level : 19
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | - | 10 | 12 |
-----
| 13 | 14 | 11 | 15 |
-----

Simpul ke-72833
Cost : 23
Level : 20
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 10 | - | 12 |
-----
| 13 | 14 | 11 | 15 |
-----

Simpul ke-72876
Cost : 23
Level : 21
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 10 | 11 | 12 |
-----
| 13 | 14 | - | 15 |
-----

Simpul ke-72929
Cost : 22
Level : 22
-----
| 1 | 2 | 3 | 4 |
-----
| 5 | 6 | 7 | 8 |
-----
| 9 | 10 | 11 | 12 |
-----
| 13 | 14 | 15 | - |
-----

Solusi ditemukan
Waktu Eksekusi : 4.755898952484131 detik

```

Pengujian test4.txt

```

Masukkan nama file : test4.txt
Tidak ada solusi yang mungkin
Waktu Eksekusi : 0.0050013065338134766 detik

```

Pengujian test5.txt

```

Masukkan nama file : test5.txt
Tidak ada solusi yang mungkin
Waktu Eksekusi : 0.0020258426666259766 detik

```

### C. Checklist Program

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

## D. Kode Program

File priorityQueue.py

```
from heapq import heappush, heappop

class Node:
    def __init__(self, parent, id, matrix, cost, block_pos, level):
        self.parent = parent
        self.id = id
        self.matrix = matrix
        self.cost = cost
        self.block_pos = block_pos
        self.level = level

    def __lt__(self, other):
        if self.cost == other.cost:
            return self.level < other.level
        return self.cost < other.cost

class PriorityQueue:
    def __init__(self):
        self.heap = []

    def push(self, k):
        heappush(self.heap, k)

    def pop(self):
        return heappop(self.heap)

    def empty(self):
        return not self.heap
```

File branchAndBound.py

```
from sympy import false, true
from priorityQueue import PriorityQueue, Node
from copy import deepcopy

pq = PriorityQueue()
id = 1
done = False

def lokasi(matriks, x) :
    for i in range(4) :
```

```

        for j in range (4) :
            if matriks[i][j] == x :
                return 4*i+j

def kurang(matriks, i) :
    count = 0
    for j in range (1, i):
        if lokasi(matriks, i) < lokasi(matriks, j):
            count += 1
    return count

def totalKurang(node) :
    sum = 0
    for i in range (1, 17):
        sum += kurang(node.matrix, i)
    if (node.block_pos[0] - node.block_pos[1]) % 2 != 0 :
        sum += 1
    return sum

def cost(node):
    count=0
    for i in range (4):
        for j in range (4):
            if node.matrix[i][j] != 4*i + j + 1 and node.matrix[i][j] != 0:
                count+=1
    node.cost = count + node.level

def isSolution(node):
    solution = true
    for i in range (4) :
        for j in range (4) :
            if(node.matrix[i][j] != 4*i+j+1) :
                solution = false
    return solution

def printNode(node):
    print("Simpul ke-" + str(node.id))
    print("Cost : " + str(node.cost))
    print("Level : " + str(node.level))
    for i in range (21) :
        print("-", end="")
    print("")
    for i in range (4) :
        for j in range (4) :
            print("|", end="")

```



```

        if(node.matrix[i][j] >= 10):
            if node.matrix[i][j] == 16 :
                print("  -", end=" ")
            else :
                print(" " + str(node.matrix[i][j]), end=" ")
        else :
            print(" " + str(node.matrix[i][j]), end=" ")
    print("|", end="\n")
    for i in range (21) :
        print("-", end="")
    print("")
    print("")

def printSolution(node):
    listNode = []
    while node.parent != None :
        listNode.append(node)
        node = node.parent
    listNode.append(node)
    for i in range (len(listNode)-1, -1, -1):
        printNode(listNode[i])
    done = true

def readFile(fileName):
    dir = ".\\test\\"
    dir += fileName
    file = open(dir, "r")
    matriks = [[int(num) for num in line.split(' ')] for line in file]
    return matriks

def nextNode(node):
    global id
    global pq
    if node.block_pos[0] != 0 and (node.parent == None or (node.parent).block_pos
!= [node.block_pos[0]-1, node.block_pos[1]]):
        id+=1
        newNodeMatrix1 = deepcopy(node.matrix)
        newNodeMatrix1[node.block_pos[0]][node.block_pos[1]] =
node.matrix[node.block_pos[0]-1][node.block_pos[1]]
        newNodeMatrix1[node.block_pos[0]-1][node.block_pos[1]] = 16
        newNode = Node(node, id, newNodeMatrix1, 0, [node.block_pos[0] - 1,
node.block_pos[1]], node.level + 1)
        cost(newNode)
        if isSolution(newNode):
            print("Solusi ditemukan")

```

```

        printSolution(newNode)
        pq.push(newNode)
        if node.block_pos[1] != 3 and (node.parent == None or (node.parent).block_pos
!= [node.block_pos[0], node.block_pos[1]+1]):
            id+=1
            newNodeMatrix2 = deepcopy(node.matrix)
            newNodeMatrix2[node.block_pos[0]][node.block_pos[1]] =
node.matrix[node.block_pos[0]][node.block_pos[1]+1]
            newNodeMatrix2[node.block_pos[0]][node.block_pos[1]+1] = 16
            newNode = Node(node, id, newNodeMatrix2, 0, [node.block_pos[0],
node.block_pos[1]+1], node.level + 1)
            cost(newNode)
            if isSolution(newNode):
                print("Solusi ditemukan")
                printSolution(newNode)
                pq.push(newNode)
            if node.block_pos[0] != 3 and (node.parent == None or (node.parent).block_pos
!= [node.block_pos[0]+1, node.block_pos[1]]):
                id+=1
                newNodeMatrix3 = deepcopy(node.matrix)
                newNodeMatrix3[node.block_pos[0]][node.block_pos[1]] =
node.matrix[node.block_pos[0]+1][node.block_pos[1]]
                newNodeMatrix3[node.block_pos[0]+1][node.block_pos[1]] = 16
                newNode = Node(node, id, newNodeMatrix3, 0, [node.block_pos[0] + 1,
node.block_pos[1]], node.level + 1)
                cost(newNode)
                if isSolution(newNode):
                    print("Solusi ditemukan")
                    printSolution(newNode)
                    pq.push(newNode)
            if node.block_pos[1] != 0 and (node.parent == None or (node.parent).block_pos
!= [node.block_pos[0], node.block_pos[1]-1]):
                id+=1
                newNodeMatrix4 = deepcopy(node.matrix)
                newNodeMatrix4[node.block_pos[0]][node.block_pos[1]] =
node.matrix[node.block_pos[0]][node.block_pos[1]-1]
                newNodeMatrix4[node.block_pos[0]][node.block_pos[1]-1] = 16
                newNode = Node(node, id, newNodeMatrix4, 0, [node.block_pos[0],
node.block_pos[1]-1], node.level + 1)
                cost(newNode)
                if isSolution(newNode):
                    print("Solusi ditemukan")
                    printSolution(newNode)
                    pq.push(newNode)

```

File main.py

```
import branchAndBound as bnb
from branchAndBound import pq, done
from priorityQueue import Node
from time import time

fileName = input("Masukkan nama file : ")

start_time = time()
initial_matrix = bnb.readFile(fileName)
for i in range (4) :
    for j in range (4) :
        if(initial_matrix[i][j] == 16) :
            initial_block_pos = [i, j]
            break
root = Node(None, 0 , initial_matrix, 0, initial_block_pos, 0)
bnb.cost(root)
pq.push(root)
if(bnb.totalKurang(root) % 2 == 0):
    while(not pq.empty() and not done):
        node = pq.pop()
        if bnb.isSolution(node):
            print("Solusi ditemukan")
            break
        bnb.nextNode(node)
else :
    print("Tidak ada solusi yang mungkin")

end_time = time()
print("Waktu Eksekusi : " + str(end_time - start_time)+ " detik")
```

#### E. Test Case Berupa Teks

test1.txt

1 2 3 4

5 6 16 8

9 10 7 11

13 14 15 12

test2.txt

1 7 2 4  
9 5 3 8  
13 16 11 12  
10 6 14 15

test3.txt  
1 2 3 4  
5 6 7 8  
16 13 9 10  
11 12 15 14

test4.txt  
1 3 4 15  
2 16 5 12  
7 6 11 14  
8 9 10 13

test5.txt  
7 2 10 9  
6 16 1 3  
4 5 12 11  
15 8 14 13

## F. Link Github

[https://github.com/ubaidalih/Tucil3\\_13520085](https://github.com/ubaidalih/Tucil3_13520085)