

Tugas Kecil 3 IF2211 Strategi Algoritma

Semester 2 Tahun 2021/2022

Penyelesaian Persoalan 15-Puzzle dengan Algoritma *Branch and Bound*



Disusun oleh

M Syahrul Surya Putra – 13520161

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

2022

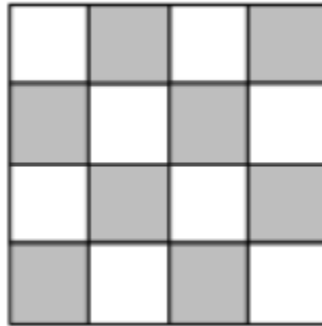
A. Algoritma Branch and Bound

Algoritma yang digunakan pada tugas kecil penyelesaian permainan 15-Puzzle ini ialah *Branch and Bound*. Adapun langkah-langkahnya adalah sebagai berikut:

1. Pertama-tama, program akan menerima input berupa nama file pada user. File merupakan sebuah test case yang berada pada folder Test dan sudah dipastikan valid angkanya. Jika file ditemukan, program akan mengeluarkan error dan otomatis keluar
2. Setelah membaca file, program akan mengubah input yang telah diambil dari file masukan user menjadi sebuah class Node
3. Selain itu, program akan menginisiasi variable waktu mulai, jumlah node, queue (prio), dan dictionary
4. Selanjutnya, program akan mengecek apakah test case yang dimasukkan oleh user itu reachable atau tidak. Cara mengeceknya ialah dengan menggunakan rumus

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

Dimana fungsi Kurang menunjukkan banyaknya ubin bernomor j, sedemikian sehingga $j < I$ dan $POSISI(j) > POSISI(i)$. Dan untuk X akan bernilai 1 jika sel kosong pada posisi awal ada pada sel yang diarsir



5. Jika hasilnya genap, program akan melanjutkan mencari jawaban. Namun, jika hasilnya ganjil, program akan langsung berhenti karena test case yang diberikan tidak ada goal yang bisa dicapai
6. Selanjutnya, program akan memasukkan node pada queue. Setiap queue, program akan mengecek apakah node solusi atau bukan
7. Ketika di loop tidak sesuai dengan solusi, node yang ada akan membangkitkan child sesuai dengan move yang tersedia. Perlu dicatat, jika child dibangkitkan oleh parent yang memberikan perintah misalnya "DOWN", child tersebut tidak bisa membangkitkan childnya sendiri dengan perintah "UP"
8. Child yang belum ada pada dict, akan dimasukkan pada queue (prio)
9. Program akan terus berjalan sampai queue tidak ada isinya atau telah ditemukan solusi

B. Source Code

- node.py

```
- import numpy as np
-
- class Node:
-     def __init__(self, parent, mtx, step, cmd=""):
-         self.parent = parent
-         self.val = mtx
-         self.move = ["UP", "RIGHT", "DOWN", "LEFT"]
-         self.takenMove = cmd
-
-         # remove ava move
-         if (cmd == "UP"):
-             self.move.remove("DOWN")
-         if (cmd == "DOWN"):
-             self.move.remove("UP")
-         if (cmd == "LEFT"):
-             self.move.remove("RIGHT")
-         if (cmd == "RIGHT"):
-             self.move.remove("LEFT")
-
-         self.step = step
-         self.cost = self.count() + self.step
-
-     def findEmpty(self):
-         for i in range(16):
-             if self.val[i] == 16:
-                 return i
-
-     def print(self):
-         for i in range(16):
-             if (self.val[i] == 16):
-                 print("   ", end="")
-             else:
-                 print("%4d" % self.val[i], end="")
-             if ((i + 1) % 4 != 0):
-                 print(" ", end="")
-             else:
-                 print()
-
-     def isSolution(self):
-         solution =
- np.array([1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16])
-
-         return (np.array_equal(self.val, solution))
```

```
-  
- # ngitung tile yang gk sesuai, exclude empty  
- def count(self):  
-     res = 0  
-  
-     for i in range(16):  
-         if self.val[i] != 16 and self.val[i] != i+1:  
-             res += 1  
-  
-     return res  
-  
- # for raisechild purposes  
- def copy(self):  
-     return np.array(self.val[:])  
-  
- # buat prioqueue  
- def __lt__(self, other):  
-     return self.cost < other.cost  
-  
- def raiseChild(self):  
-     res = []  
-     idx = self.findEmpty()  
-  
-     for move in self.move:  
-         arr = self.copy()  
-         if move == "UP":  
-             if (idx // 4) - 1 >= 0:  
-                 temp = arr[idx - 4]  
-                 arr[idx - 4] = arr[idx]  
-                 arr[idx] = temp  
-         if move == "DOWN":  
-             if (idx // 4) + 1 <= 3:  
-                 temp = arr[idx + 4]  
-                 arr[idx + 4] = arr[idx]  
-                 arr[idx] = temp  
-         if move == "LEFT":  
-             if (idx % 4) - 1 >= 0:  
-                 temp = arr[idx - 1]  
-                 arr[idx - 1] = arr[idx]  
-                 arr[idx] = temp  
-         if move == "RIGHT":  
-             if (idx % 4) + 1 <= 3:  
-                 temp = arr[idx + 1]  
-                 arr[idx + 1] = arr[idx]  
-                 arr[idx] = temp
```

```
-         res.append(Node(self, arr, self.step + 1, move))  
-  
-         return res
```

- util.py

```
import numpy as np  
  
# get kurang  
def kurang(mat):  
    res = [0 for i in range(16)]  
    print("\n-- Reachable Goal --")  
    for i in range(16):  
        for j in range(i+1, 16):  
            if mat.val[i] > mat.val[j]:  
                res[mat.val[i]-1] += 1  
    print("Kurang(%d) = %d" % (mat.val[i], res[mat.val[i]-1]))  
  
    return res  
  
# get sumkurang sama x  
def sum_kurang(mat):  
    sumRes = sum(kurang(mat))  
    idx = mat.findEmpty()  
  
    if ((idx // 4 + idx % 4) % 2 == 0):  
        X = 0  
    else:  
        X = 1  
  
    print("Sum Kurang(i) + X = %d" % (sumRes + X))  
    return (sumRes + X)  
  
def logo():  
    print("                                ")  
    print(" / | ____ |      | _ \\ \\ _ _ _ _ _ _ _ | | ____ ")  
    print(" | | ____ \\ \\ _ _ _ _ | | ) | | | | _ / _ / | / _ \\ \\ ")  
    print(" | | ____ ) | _ _ _ _ | _ / | | | | / / / / | | _ / ")  
    print(" | | ____ /      | _ | _ \\ _ , / _ _ / _ _ | | _ _ | \\n")
```

- main.py

```
- import time  
- import numpy as np  
- from queue import PriorityQueue  
- from node import Node
```

```

from util import *

logo()

try:
    fileName = input("Input File (Ex: 1.txt)\n> ")
    mat = np.loadtxt("../test/" + fileName, dtype=int).flatten()
except FileNotFoundError:
    print("\nFile not found")
    input("Press ENTER to exit...")
    exit()

start = time.time()

# Inisiasi variabel
raisedNodeCount = 0
queue = PriorityQueue()
table = {}

first = Node(None, mat, 0)
node = first
table[str(node.val)] = True

print("\n-- Initial State --")
first.print()
reach = sum_kurang(node)

if (reach % 2 != 0):
    print("\nPuzzle doesn't have a solution")
    input("Press ENTER to exit...")
else:
    queue.put((first.cost, first))

    while (not queue.empty() and not node.isSolution()):
        node = queue.get()[1]
        table[str(node.val)] = True

        nodes = node.raiseChild()
        for x in nodes:
            if (str(x.val) not in table):
                queue.put((x.cost, x))
                table[str(x.val)] = True
                raisedNodeCount += 1

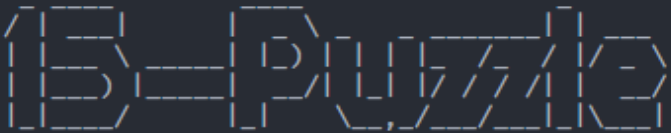
    step = []

```

```
- # Pake parent biar gk ribet conditional pas proses
- while (node.parent != None):
-     step.append(node)
-     node = node.parent
-
- for i in range(len(step)-1, -1, -1):
-     print("\n-- Step %d: %s --" % (len(step) - i,
step[i].takenMove))
-     step[i].print()
-
- if (len(step) == 0):
-     stepCount = 0
- else:
-     stepCount = step[0].step
-
- print("\n-- Solution Found --")
- print("Step Taken: %d" % stepCount)
- print("Raised Node Count: %d" % raisedNodeCount)
- print("Time Taken: %f seconds" % (time.time() - start))
```

C. Test Case

- succ1.txt

Input	Output
<pre> 5 1 3 4 9 2 7 8 16 6 15 11 13 10 14 12 </pre>	 <pre> Input File (Ex: 1.txt) > succ1.txt -- Initial State -- 5 1 3 4 9 2 7 8 - 6 15 11 13 10 14 12 -- Reachable Goal -- Kurang(5) = 4 Kurang(1) = 0 Kurang(3) = 1 Kurang(4) = 1 Kurang(9) = 4 Kurang(2) = 0 Kurang(7) = 1 Kurang(8) = 1 Kurang(16) = 7 Kurang(6) = 0 Kurang(15) = 5 Kurang(11) = 1 Kurang(13) = 2 Kurang(10) = 0 Kurang(14) = 1 Kurang(12) = 0 Sum Kurang(i) + X = 28 -- Step 1: UP -- 5 1 3 4 - 2 7 8 9 6 15 11 13 10 14 12 -- Step 2: UP -- - 1 3 4 5 2 7 8 9 6 15 11 13 10 14 12 -- Step 3: RIGHT -- 1 - 3 4 5 2 7 8 9 6 15 11 13 10 14 12 </pre>


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

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

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

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

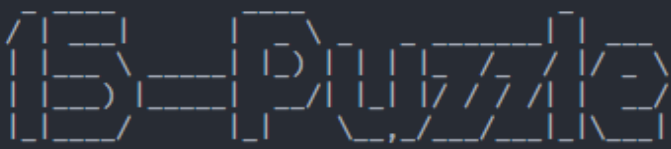
```
-- Step 8: UP --  
 1  2  3  4  
 5  6  7  8  
 9 10  - 11  
13 14 15 12
```

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

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

```
-- Solution Found --  
Step Taken: 10  
Raised Node Count: 24  
Time Taken: 0.020944 seconds
```

- succ2.txt

Input	Output
<pre> 1 6 2 4 5 16 3 8 9 7 15 11 13 14 10 12 </pre>	 <pre> Input File (Ex: 1.txt) > succ2.txt -- Initial State -- 1 6 2 4 5 - 3 8 9 7 15 11 13 14 10 12 -- Reachable Goal -- Kurang(1) = 0 Kurang(6) = 4 Kurang(2) = 0 Kurang(4) = 1 Kurang(5) = 1 Kurang(16) = 10 Kurang(3) = 0 Kurang(8) = 1 Kurang(9) = 1 Kurang(7) = 0 Kurang(15) = 5 Kurang(11) = 1 Kurang(13) = 2 Kurang(14) = 2 Kurang(10) = 0 Kurang(12) = 0 Sum Kurang(i) + X = 28 -- Step 1: LEFT -- 1 6 2 4 - 5 3 8 9 7 15 11 13 14 10 12 -- Step 2: DOWN -- 1 6 2 4 9 5 3 8 - 7 15 11 13 14 10 12 -- Step 3: DOWN -- 1 6 2 4 9 5 3 8 13 7 15 11 - 14 10 12 </pre>

		<pre> -- Step 4: RIGHT -- 1 6 2 4 9 5 3 8 13 7 15 11 14 - 10 12 -- Step 5: RIGHT -- 1 6 2 4 9 5 3 8 13 7 15 11 14 10 - 12 -- Step 6: UP -- 1 6 2 4 9 5 3 8 13 7 - 11 14 10 15 12 -- Step 7: LEFT -- 1 6 2 4 9 5 3 8 13 - 7 11 14 10 15 12 -- Step 8: DOWN -- 1 6 2 4 9 5 3 8 13 10 7 11 14 - 15 12 -- Step 9: LEFT -- 1 6 2 4 9 5 3 8 13 10 7 11 - 14 15 12 -- Step 10: UP -- 1 6 2 4 9 5 3 8 - 10 7 11 13 14 15 12 -- Step 11: UP -- 1 6 2 4 - 5 3 8 9 10 7 11 13 14 15 12 -- Step 12: RIGHT -- 1 6 2 4 5 - 3 8 9 10 7 11 13 14 15 12 </pre>	
--	--	---	--

-- Step 13: UP --

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

-- Step 14: RIGHT --

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

-- Step 15: DOWN --

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

-- Step 16: DOWN --

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

-- Step 17: RIGHT --

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

-- Step 18: DOWN --

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

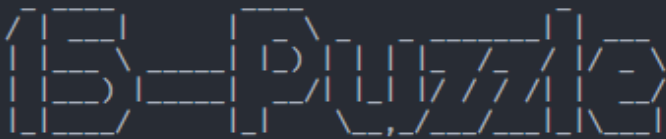
-- Solution Found --

Step Taken: 18

Raised Node Count: 6462

Time Taken: 1.049055 seconds

- succ3.txt

Input	Output
<pre> 5 2 8 10 1 11 6 4 7 9 16 3 13 14 15 12 </pre>	 <pre> Input File (Ex: 1.txt) > succ3.txt -- Initial State -- 5 2 8 10 1 11 6 4 7 9 - 3 13 14 15 12 -- Reachable Goal -- Kurang(5) = 4 Kurang(2) = 1 Kurang(8) = 5 Kurang(10) = 6 Kurang(1) = 0 Kurang(11) = 5 Kurang(6) = 2 Kurang(4) = 1 Kurang(7) = 1 Kurang(9) = 1 Kurang(16) = 5 Kurang(3) = 0 Kurang(13) = 1 Kurang(14) = 1 Kurang(15) = 1 Kurang(12) = 0 Sum Kurang(i) + X = 34 -- Step 1: UP -- 5 2 8 10 1 11 - 4 7 9 6 3 13 14 15 12 -- Step 2: LEFT -- 5 2 8 10 1 - 11 4 7 9 6 3 13 14 15 12 -- Step 3: LEFT -- 5 2 8 10 - 1 11 4 7 9 6 3 13 14 15 12 </pre>

		<pre> -- Step 4: DOWN -- 5 2 8 10 7 1 11 4 - 9 6 3 13 14 15 12 -- Step 5: RIGHT -- 5 2 8 10 7 1 11 4 9 - 6 3 13 14 15 12 -- Step 6: RIGHT -- 5 2 8 10 7 1 11 4 9 6 - 3 13 14 15 12 -- Step 7: UP -- 5 2 8 10 7 1 - 4 9 6 11 3 13 14 15 12 -- Step 8: RIGHT -- 5 2 8 10 7 1 4 - 9 6 11 3 13 14 15 12 -- Step 9: UP -- 5 2 8 - 7 1 4 10 9 6 11 3 13 14 15 12 -- Step 10: LEFT -- 5 2 - 8 7 1 4 10 9 6 11 3 13 14 15 12 -- Step 11: LEFT -- 5 - 2 8 7 1 4 10 9 6 11 3 13 14 15 12 </pre>	
--	--	--	--

		<pre>-- Step 12: DOWN -- 5 1 2 8 7 - 4 10 9 6 11 3 13 14 15 12 -- Step 13: LEFT -- 5 1 2 8 - 7 4 10 9 6 11 3 13 14 15 12 -- Step 14: UP -- - 1 2 8 5 7 4 10 9 6 11 3 13 14 15 12 -- Step 15: RIGHT -- 1 - 2 8 5 7 4 10 9 6 11 3 13 14 15 12 -- Step 16: RIGHT -- 1 2 - 8 5 7 4 10 9 6 11 3 13 14 15 12 -- Step 17: DOWN -- 1 2 4 8 5 7 - 10 9 6 11 3 13 14 15 12 -- Step 18: RIGHT -- 1 2 4 8 5 7 10 - 9 6 11 3 13 14 15 12 -- Step 19: DOWN -- 1 2 4 8 5 7 10 3 9 6 11 - 13 14 15 12</pre>	
--	--	--	--

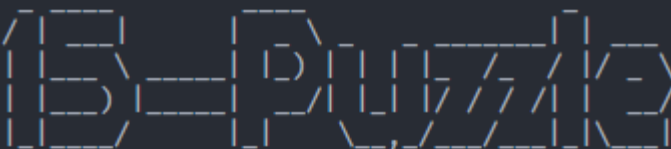
		<pre> -- Step 20: LEFT -- 1 2 4 8 5 7 10 3 9 6 - 11 13 14 15 12 -- Step 21: UP -- 1 2 4 8 5 7 - 3 9 6 10 11 13 14 15 12 -- Step 22: RIGHT -- 1 2 4 8 5 7 3 - 9 6 10 11 13 14 15 12 -- Step 23: UP -- 1 2 4 - 5 7 3 8 9 6 10 11 13 14 15 12 -- Step 24: LEFT -- 1 2 - 4 5 7 3 8 9 6 10 11 13 14 15 12 -- Step 25: DOWN -- 1 2 3 4 5 7 - 8 9 6 10 11 13 14 15 12 -- Step 26: LEFT -- 1 2 3 4 5 - 7 8 9 6 10 11 13 14 15 12 -- Step 27: DOWN -- 1 2 3 4 5 6 7 8 9 - 10 11 13 14 15 12 -- Step 28: RIGHT -- 1 2 3 4 5 6 7 8 9 10 - 11 13 14 15 12 </pre>	
--	--	---	--

		<pre> -- Step 29: RIGHT -- 1 2 3 4 5 6 7 8 9 10 11 - 13 14 15 12 -- Step 30: DOWN -- 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 - -- Solution Found -- Step Taken: 30 Raised Node Count: 1898994 Time Taken: 344.422530 seconds </pre>
--	--	---

- fail1.txt

Input	Output
<pre> 5 2 8 10 1 16 6 4 7 9 11 3 13 14 15 12 </pre>	<pre> 15-Puzzle Input File (Ex: 1.txt) > fail1.txt -- Initial State -- 5 2 8 10 1 - 6 4 7 9 11 3 13 14 15 12 -- Reachable Goal -- Kurang(5) = 4 Kurang(2) = 1 Kurang(8) = 5 Kurang(10) = 6 Kurang(1) = 0 Kurang(16) = 10 Kurang(6) = 2 Kurang(4) = 1 Kurang(7) = 1 Kurang(9) = 1 Kurang(11) = 1 Kurang(3) = 0 Kurang(13) = 1 Kurang(14) = 1 Kurang(15) = 1 Kurang(12) = 0 Sum Kurang(i) + X = 35 Puzzle doesn't have a solution Press ENTER to exit... </pre>

- fail2.txt

Input	Output
<pre> 5 1 3 4 9 2 7 8 10 6 15 11 13 16 14 12 </pre>	 <pre> Input File (Ex: 1.txt) > fail2.txt -- Initial State -- 5 1 3 4 9 2 7 8 10 6 15 11 13 - 14 12 -- Reachable Goal -- Kurang(5) = 4 Kurang(1) = 0 Kurang(3) = 1 Kurang(4) = 1 Kurang(9) = 4 Kurang(2) = 0 Kurang(7) = 1 Kurang(8) = 1 Kurang(10) = 1 Kurang(6) = 0 Kurang(15) = 4 Kurang(11) = 0 Kurang(13) = 1 Kurang(16) = 2 Kurang(14) = 1 Kurang(12) = 0 Sum Kurang(i) + X = 21 Puzzle doesn't have a solution Press ENTER to exit... </pre>

D. Lain-lain

Link Repository: <https://github.com/msyahrulsp/15-puzzle-solver>

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		√