

IF2211 Strategi Algoritma
LAPORAN TUGAS KECIL 3
Penyelesaian Persoalan 15-Puzzle
dengan
Algoritma Branch and Bound



Dosen : Rinaldi Munir

Oleh

Jonathan Yudi Gunawan

13518084

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2019/2020

DAFTAR ISI

DAFTAR ISI	2
BAB I	
DESKRIPSI MASALAH	3
1.1 Deskripsi Masalah	3
1.2 Masukan	3
1.3 Deskripsi Program	3
1.4 Keluaran	4
BAB II	
IMPLEMENTASI PROGRAM	5
2.1 Algoritma Branch and Bound	5
2.2 Struktur Program	6
BAB III	
EKSPERIMEN	7
3.1 Solvable Easy	7
3.2 Solvable Medium	9
3.3 Solvable Sample	11
3.4 Unsolvable 1	13
3.5 Unsolvable 2	14
3.6 (Bonus) Solvable Rectangle	15
BAB IV	
LAIN-LAIN	17
4.1 Spesifikasi Komputer	17
4.2 Checklist Implementasi	17
DAFTAR REFERENSI	17

BAB I

DESKRIPSI MASALAH

1.1 Deskripsi Masalah

Buatlah program dalam Python untuk menyelesaikan persoalan 15-Puzzle dengan menggunakan Algoritma Branch and Bound seperti pada materi kuliah. Nilai bound tiap simpul adalah penjumlahan cost yang diperlukan untuk sampai suatu simpul x dari akar, dengan taksiran cost simpul x untuk sampai ke goal. Taksiran cost yang digunakan adalah jumlah ubin tidak kosong yang tidak berada pada tempat sesuai susunan akhir (goal state). Untuk semua instansiasi persoalan 15-puzzle, susunan akhir yang diinginkan sesuai dengan Gambar 1.

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

Gambar 1. Susunan Akhir persoalan 15-puzzle

1.2 Masukan

Matriks yang merepresentasikan posisi awal suatu instansiasi persoalan 15-puzzle. Matriks dibaca dari berkas teks.

1.3 Deskripsi Program

Program harus dapat menentukan apakah posisi awal suatu masukan dapat diselesaikan hingga mencapai susunan akhir, dengan mengimplementasikan fungsi Kurang(i) dan posisi ubin kosong di kondisi awal (X), seperti pada materi kuliah. Jika posisi awal tidak bisa mencapai susunan akhir, program akan menampilkan pesan tidak bisa diselesaikan,. Jika dapat diselesaikan, program dapat menampilkan urutan matriks rute (path) aksi yang dilakukan dari posisi awal ke susunan akhir. Sebagai contoh pada Gambar 2, matriks yang ditampilkan ke layar adalah matriks pada simpul 1, simpul 4, simpul 10 dan simpul 23.

1.4 Keluaran

1. Matriks posisi awal.
2. Nilai dari fungsi Kurang (i) untuk setiap ubin tidak kosong pada posisi awal (nilai ini tetap dikeluarkan, baik persoalan bisa diselesaikan atau tidak bisa diselesaikan).
3. Nilai dari
4. Jika persoalan tidak dapat diselesaikan (berdasarkan hasil butir 2) keluar pesan.
5. Jika persoalan dapat diselesaikan (berdasarkan hasil butir 2), menampilkan urutan matriks seperti pada penjelasan sebelumnya.
6. Waktu eksekusi
7. Jumlah simpul yang dibangkitkan dalam pohon ruang status.

BAB II

IMPLEMENTASI PROGRAM

2.1 Algoritma Branch and Bound

Pertama, akan dihitung nilai fungsi $Kurang(i)$ untuk masing-masing petak pada papan. Fungsi $Kurang(i)$ didefinisikan sebagai banyaknya ubin bernomor j sedemikian sehingga $j < i$ dan $POSISI(j) > POSISI(i)$.

Fungsi $POSISI(i)$ didefinisikan sebagai posisi ubin bernomor i pada susunan yang diperiksa.

Kemudian akan dihitung nilai X yaitu $(\text{row petak kosong} + \text{column petak kosong}) \bmod 2$.

Setelah itu, jika jumlah semua nilai fungsi $Kurang(i)$ untuk semua petak ditambah dengan X adalah genap, maka petak dapat diselesaikan.

The proof is left as an exercise to the reader.

Untuk menyelesaikan puzzle yang dapat diselesaikan, dilakukan algoritma branch and bound dengan fungsi $cost = c(i) = f(i) + g(i)$

Fungsi $f(i)$ didefinisikan sebagai ongkos mencapai simpul i dari akar

Fungsi $g(i)$ didefinisikan sebagai ongkos mencapai simpul tujuan dari simpul i

Simpul i adalah simpul saat ini sedangkan simpul tujuan seperti pada gambar 1.

Nilai $g(i)$ merupakan nilai taksiran panjang lintasan terpendek dari simpul i ke simpul tujuan yang nilainya didefinisikan sebagai jumlah ubin tidak kosong yang tidak terdapat pada susunan akhir.

Untuk algoritma branch and bound sendiri digunakan struktur data `heapq` yang berfungsi seperti `priority queue` pada C++. Untuk tiap iterasi dilakukan penambahan state simpul yang dapat dicapai dari state saat ini (gerak ke kiri, kanan, atas, atau pun bawah) ke dalam antrian beserta nilai $c(i)$ dari state itu. Kemudian diambil simpul dengan nilai $c(i)$ terendah untuk dievaluasi selanjutnya.

Iterasi dilakukan hingga solusi ditemukan, lalu hentikan pencarian dan lakukan runut balik menuju akar untuk mengeluarkan solusinya.

2.2 Struktur Program

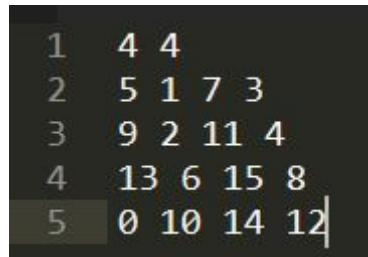
```
> tree /f
```

```
|  README.md
|
|--- doc
|--- src
|    |  main.py
|    |  solver.py
|    |  util.py
|
|--- test
|    |  solve_easy.txt
|    |  solve_medium.txt
|    |  solve_rect.txt
|    |  solve_sample.txt
|    |  unsolve_1.txt
|    |  unsolve_2.txt
```

BAB III

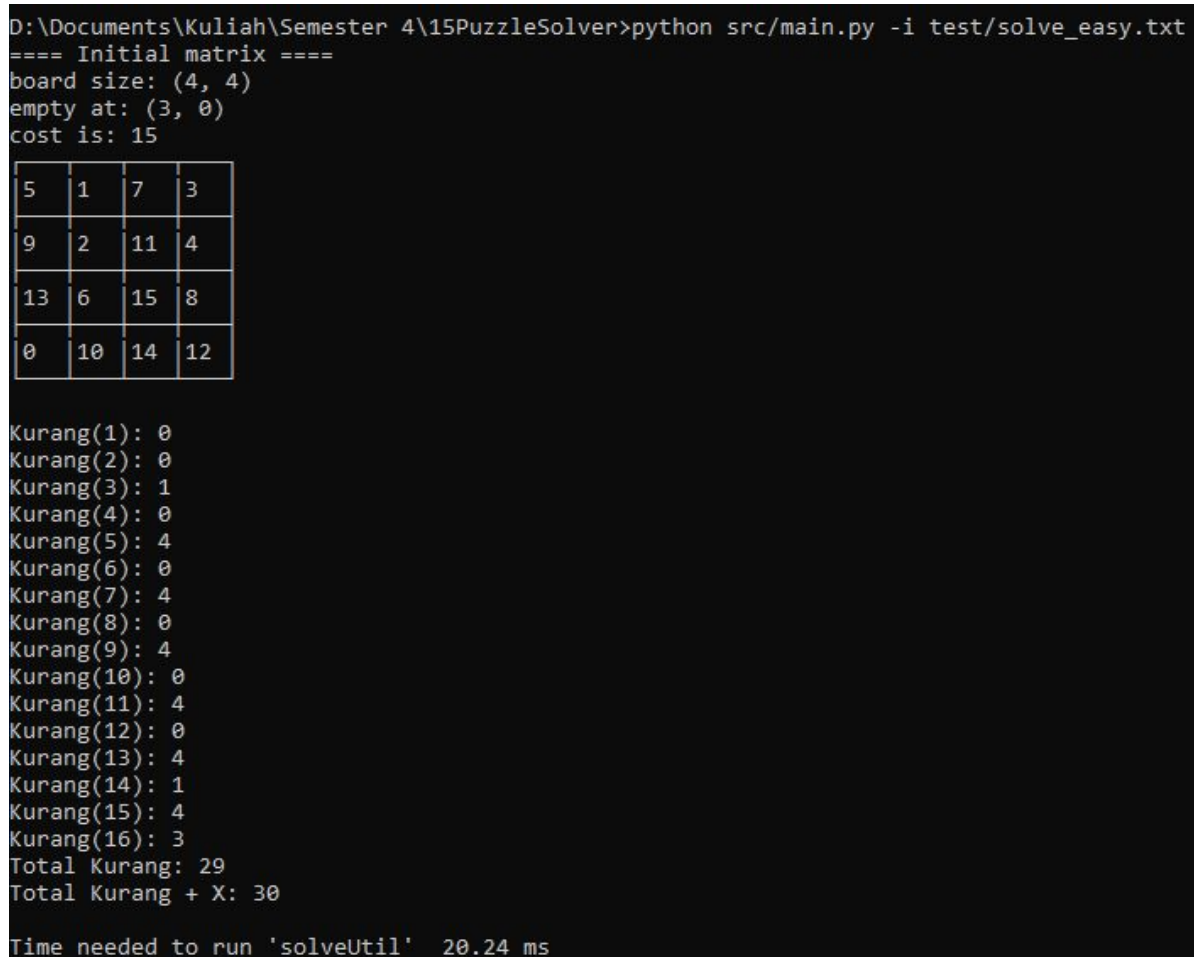
EKSPERIMEN

3.1 Solvable Easy



1	4	4		
2	5	1	7	3
3	9	2	11	4
4	13	6	15	8
5	0	10	14	12

Gambar 3.1.1 Contoh Input Test Case Solvable 1



```
D:\Documents\Kuliah\Semester 4\15PuzzleSolver>python src/main.py -i test/solve_easy.txt
==== Initial matrix ====
board size: (4, 4)
empty at: (3, 0)
cost is: 15
```

5	1	7	3
9	2	11	4
13	6	15	8
0	10	14	12

```
Kurang(1): 0
Kurang(2): 0
Kurang(3): 1
Kurang(4): 0
Kurang(5): 4
Kurang(6): 0
Kurang(7): 4
Kurang(8): 0
Kurang(9): 4
Kurang(10): 0
Kurang(11): 4
Kurang(12): 0
Kurang(13): 4
Kurang(14): 1
Kurang(15): 4
Kurang(16): 3
Total Kurang: 29
Total Kurang + X: 30
Time needed to run 'solveUtil' 20.24 ms
```

Gambar 3.1.2 Contoh Output Test Case Solvable 1

===== Steps =====			
Step 0:			
5	1	7	3
9	2	11	4
13	6	15	8
0	10	14	12
Step 1:			
5	1	7	3
9	2	11	4
0	6	15	8
13	10	14	12
Step 2:			
5	1	7	3
0	2	11	4
9	6	15	8
13	10	14	12
Step 13:			
1	2	3	4
5	6	7	0
9	10	11	8
13	14	15	12
Step 14:			
1	2	3	4
5	6	7	8
9	10	11	0
13	14	15	12
Step 15:			
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	0
33 nodes generated to solve the puzzle			

Gambar 3.1.3 Contoh Output Test Case Solvable 1 (cont.)

3.2 Solvable Medium

1	4	4		
2	1	2	3	4
3	5	6	7	0
4	9	10	12	8
5	11	13	14	15

Gambar 3.2.1 Contoh Input Test Case Solvable 2

```
D:\Documents\Kuliah\Semester 4\15PuzzleSolver>python src/main.py -i test/solve_medium.txt
==== Initial matrix ====
board size: (4, 4)
empty at: (1, 3)
cost is: 6
```

1	2	3	4
5	6	7	0
9	10	12	8
11	13	14	15

```
Kurang(1): 0
Kurang(2): 0
Kurang(3): 0
Kurang(4): 0
Kurang(5): 0
Kurang(6): 0
Kurang(7): 0
Kurang(8): 0
Kurang(9): 1
Kurang(10): 1
Kurang(11): 0
Kurang(12): 2
Kurang(13): 0
Kurang(14): 0
Kurang(15): 0
Kurang(16): 8
Total Kurang: 12
Total Kurang + X: 12

Time needed to run 'solveUtil' 3268.58 ms
```

Gambar 3.2.2 Contoh Output Test Case Solvable 2

===== Steps =====				5	6	7	8
Step 0:				9	0	11	12
1	2	3	4	13	10	14	15
5	6	7	0				
9	10	12	8	Step 16:			
11	13	14	15	1	2	3	4
Step 1:				5	6	7	8
1	2	3	4	9	10	11	12
5	6	7	8	13	0	14	15
9	10	12	0	Step 17:			
11	13	14	15	1	2	3	4
Step 2:				5	6	7	8
1	2	3	4	9	10	11	12
5	6	7	8	13	14	0	15
9	10	0	12	Step 18:			
11	13	14	15	1	2	3	4
Step 3:				5	6	7	8
1	2	3	4	9	10	11	12
5	6	0	8	13	14	15	0
9	10	7	12	3899 nodes generated to solve the puzzle			
11	13	14	15				

Gambar 3.2.3 Contoh Output Test Case Solvable 2 (cont.)

3.3 Solvable Sample

1	4	4	
2	1	2	3 4
3	5	6	0 8
4	9	10	7 11
5	13	14	15 12

Gambar 3.3.1 Contoh Input Test Case Solvable 3

```
D:\Documents\Kuliah\Semester 4\15PuzzleSolver>python src/main.py -i test/solve_sample.txt
==== Initial matrix ====
board size: (4, 4)
empty at: (1, 2)
cost is: 3
```

1	2	3	4
5	6	0	8
9	10	7	11
13	14	15	12

```
Kurang(1): 0
Kurang(2): 0
Kurang(3): 0
Kurang(4): 0
Kurang(5): 0
Kurang(6): 0
Kurang(7): 0
Kurang(8): 1
Kurang(9): 1
Kurang(10): 1
Kurang(11): 0
Kurang(12): 0
Kurang(13): 1
Kurang(14): 1
Kurang(15): 1
Kurang(16): 9
Total Kurang: 15
Total Kurang + X: 16

Time needed to run 'solveUtil' 0.00 ms
```

Gambar 3.3.2 Contoh Output Test Case Solvable 3

===== Steps =====

Step 0:

1	2	3	4
5	6	0	8
9	10	7	11
13	14	15	12

Step 1:

1	2	3	4
5	6	7	8
9	10	0	11
13	14	15	12

Step 2:

1	2	3	4
5	6	7	8
9	10	11	0
13	14	15	12

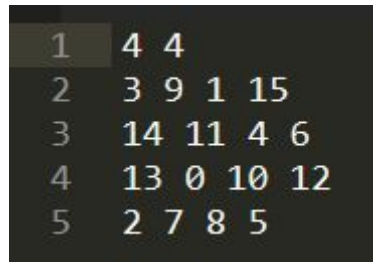
Step 3:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	0

10 nodes generated to solve the puzzle

Gambar 3.3.3 Contoh Output Test Case Solvable 3 (cont.)

3.4 Unsolvable 1



1	4	4		
2	3	9	1	15
3	14	11	4	6
4	13	0	10	12
5	2	7	8	5

Gambar 3.4.1 Contoh Input Test Case Unsolvable 1

```
D:\Documents\Kuliah\Semester 4\15PuzzleSolver>python src/main.py -i test/unsolve_1.txt
==== Initial matrix ====
board size: (4, 4)
empty at: (2, 1)
cost is: 14
```

3	9	1	15
14	11	4	6
13	0	10	12
2	7	8	5

```
Kurang(1): 0
Kurang(2): 0
Kurang(3): 2
Kurang(4): 1
Kurang(5): 0
Kurang(6): 2
Kurang(7): 1
Kurang(8): 1
Kurang(9): 7
Kurang(10): 4
Kurang(11): 7
Kurang(12): 4
Kurang(13): 6
Kurang(14): 10
Kurang(15): 11
Kurang(16): 6
Total Kurang: 62
Total Kurang + X: 63
Puzzle is unsolvable!
```

Gambar 3.4.2 Contoh Output Test Case Unsolvable 1

3.5 Unsolvable 2

1	4	4	
2	1	6	3 2
3	5	7	4 0
4	9	10	11 8
5	13	14	15 12

Gambar 3.5.1 Contoh Input Test Case Unsolvable 2

```
D:\Documents\Kuliah\Semester 4\15PuzzleSolver>python src/main.py -i test/unsolve_2.txt
==== Initial matrix ====
board size: (4, 4)
empty at: (1, 3)
cost is: 6
```

1	6	3	2
5	7	4	0
9	10	11	8
13	14	15	12

```
Kurang(1): 0
Kurang(2): 0
Kurang(3): 1
Kurang(4): 0
Kurang(5): 1
Kurang(6): 4
Kurang(7): 1
Kurang(8): 0
Kurang(9): 1
Kurang(10): 1
Kurang(11): 1
Kurang(12): 0
Kurang(13): 1
Kurang(14): 1
Kurang(15): 1
Kurang(16): 8
Total Kurang: 21
Total Kurang + X: 21
Puzzle is unsolvable!
```

Gambar 3.5.2 Contoh Output Test Case Unsolvable 2

3.6 (Bonus) Solvable Rectangle

1	2	4		
2	1	6	2	7
3	5	0	4	3

Gambar 3.6.1 Contoh Input Test Case Solvable Rectangle

```
D:\Documents\Kuliah\Semester 4\15PuzzleSolver>python src/main.py -i test/solve_rect.txt
==== Initial matrix ====
board size: (2, 4)
empty at: (1, 1)
cost is: 6
```

1	6	2	7
5	0	4	3

```
Kurang(1): 0
Kurang(2): 0
Kurang(3): 0
Kurang(4): 1
Kurang(5): 2
Kurang(6): 4
Kurang(7): 3
Kurang(8): 2
Total Kurang: 12
Total Kurang + X: 12

Time needed to run 'solveUtil' 20.44 ms
```

Gambar 3.6.2 Contoh Output Test Case Solvable Rectangle

```
===== Steps =====
Step 0:


|   |   |   |   |
|---|---|---|---|
| 1 | 6 | 2 | 7 |
| 5 | 0 | 4 | 3 |



Step 1:


|   |   |   |   |
|---|---|---|---|
| 1 | 0 | 2 | 7 |
| 5 | 6 | 4 | 3 |



Step 2:


|   |   |   |   |
|---|---|---|---|
| 1 | 2 | 0 | 7 |
| 5 | 6 | 4 | 3 |



Step 3:


|   |   |   |   |
|---|---|---|---|
| 1 | 2 | 4 | 7 |
| 5 | 6 | 0 | 3 |



Step 4:


|   |   |   |   |
|---|---|---|---|
| 1 | 2 | 4 | 7 |
| 5 | 6 | 3 | 0 |



Step 4:


|   |   |   |   |
|---|---|---|---|
| 1 | 2 | 4 | 7 |
| 5 | 6 | 3 | 0 |



Step 5:


|   |   |   |   |
|---|---|---|---|
| 1 | 2 | 4 | 0 |
| 5 | 6 | 3 | 7 |



Step 6:


|   |   |   |   |
|---|---|---|---|
| 1 | 2 | 0 | 4 |
| 5 | 6 | 3 | 7 |



Step 7:


|   |   |   |   |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 5 | 6 | 0 | 7 |



Step 8:


|   |   |   |   |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 0 |



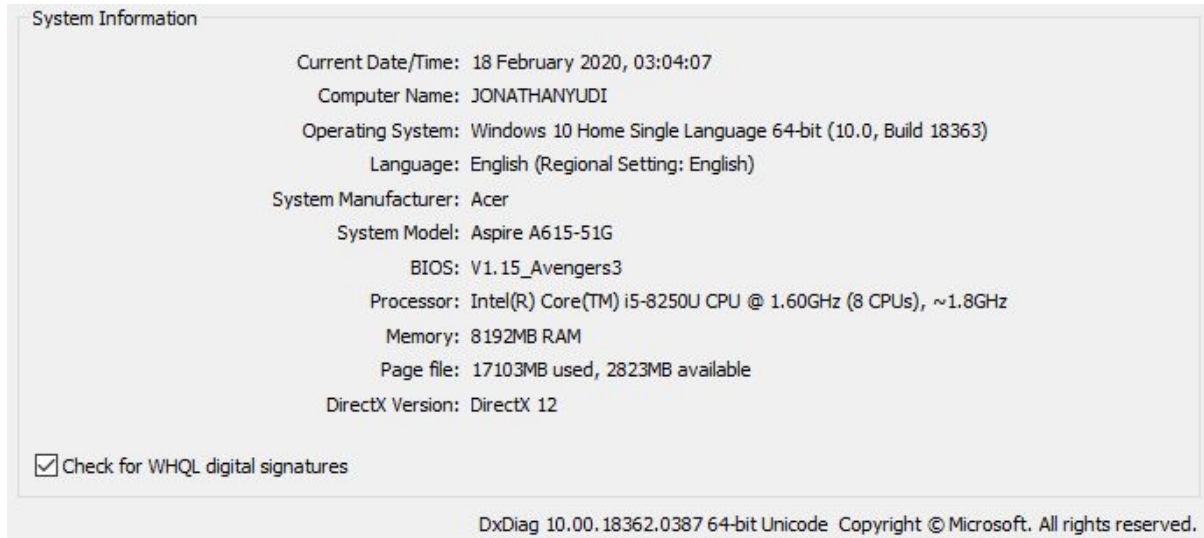
108 nodes generated to solve the puzzle
```

Gambar 3.6.3 Contoh Output Test Case Solvable Rectangle (cont.)

BAB IV

LAIN-LAIN

4.1 Spesifikasi Komputer



4.2 Checklist Implementasi

No	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	

* untuk kasus tertentu (papan cukup teracak) pencarian solusi dapat memakan waktu hingga 1 jam (mungkin lebih)

DAFTAR REFERENSI

[http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2019-2020/Tugas-Kecil-3-\(2020\).pdf](http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2019-2020/Tugas-Kecil-3-(2020).pdf)

[http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Branch-&-Bound-\(2018\).pdf](http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Branch-&-Bound-(2018).pdf)