**TUGAS KECIL 03**

**PENYELESAIAN PERSOALAN 15-PUZZLE DENGAN**

**ALGORITMA BRANCH AND BOUND**

**IF2211 – STRATEGI ALGORITMA**

A picture containing text, clipart

Description automatically generated

**Disusun oleh:**

13520034 - Bryan Bernigen

**Institut Teknologi Bandung**

**Jl. Ganesha NO. 10, Bandung 40132**

**2022**

Daftar Isi

[Cara Kerja Algoritma Branch and Bound 3](#_Toc99817329)

[Algoritma Branch and Bound Dengan Visualisasi 4](#_Toc99817330)

[Source Code 8](#_Toc99817331)

[PriotizedItem.py 8](#_Toc99817332)

[Gui.py 8](#_Toc99817333)

[fifteen\_puzzle.py 10](#_Toc99817334)

[main.py 15](#_Toc99817335)

[Hasil Pengujian 17](#_Toc99817336)

[1. Berhasil1.txt 17](#_Toc99817337)

[2. Berhasil2.txt 17](#_Toc99817338)

[3. Berhasil3.txt 19](#_Toc99817339)

[4. Gagal1.txt 20](#_Toc99817340)

[5. Gagal2.txt 21](#_Toc99817341)

[Lampiran 22](#_Toc99817342)

# Cara Kerja Algoritma Branch and Bound

Part 1: Penghitungan Nilai Pembatas

1. Perhitungan 1:

Evaluasi setiap nilai yang ada (1...16) apakah terdapat nilai lain yang lebih kecil dari nilai yang sedang dievaluasi, namun lokasinya berada pada indeks yang lebih besar. Hitung jumlah seluruh nilai yang lokasinya tidak tepat.

1. Perhitungan 2:

Khusus untuk langkah pertama (akarnya), hitung juga nilai lokasi nilai 16 (kotak kosong) dengan rumus sebagai berikut: 1 jika kotak kosong berada pada indeks 1,3,4,6 + 8i, i=0,1. Dan 0 jika kotak kosong berada pada indeks 0,2,5,7 + 8i, i=0,1. Untuk visualisasi lebih jelas silakan mengacu pada bab selanjutnya

1. Jika hasil perhitungan 1 + hasil perhitungan 2 bernilai ganjil, maka puzzle tersebut tidak dapat diselesaikan. Jika hasil perhitungan 1 dan perhitungan 2 bernilai genap, maka puzzle akan dikerjakan. Awali dengan memasukkan akar ke queue prioritas lalu ikuti langkah-langkah pada part 2.

Part 2: Branch and Bound

1. Ambil nilai pertama yang ada pada antrean (nilai diurut berdasarkan nilai terkecil yang didapat pada perhitungan 1 part 1).
2. Bentuk semua pergerakan (atas, bawah, kiri, kanan) yang mungkin dari keadaan tersebut. Pergerakan dibentuk jika memenuhi 2 syarat, yakni:
   1. Tidak menghasilkan keadaan yang sama dengan ranting pemanggilnya (contoh: command LEFT tidak akan diikuti oleh command RIGHT karena kedua command tersebut hanya akan bolak-balik menghasilkan keadaan yang sama)
   2. Tidak membuat kotak kosong menjadi out of bounds. (contoh: ketika kotak kosong berada di kanan atas, command UP dan RIGHT tidak akan dipanggil)
   3. (Opsional untuk optimasi) cek apakah sate tersebut sudah ada di hash map atau belu. Jika belum ada, state tersebut akan dibangkitkan dan dibuat true di hash map.
3. Untuk setiap command yang dihasilkan, hitung jumlah nilai yang berada tidak pada tempatnya. Setelah itu, masukkan hasil pergerakan tersebut ke queue prioritas dengan nilai perhitungan ditambah kedalaman node tersebut sebagai prioritasnya (semakin sedikit nilai yang tidak pada tempatnya, semakin besar prioritasnya)
4. Jika hasil perhitungan nilai tidak pada tempatnya= 0, maka solusi ditemukan.
5. Jika solusi belum ditemukan, lakukan langkah 1-5 part 2 sampai solusi ditemukan karena berdasarkan perhitungan, solusi dapat dipastikan ada. (tidak perlu menunggu queue kosong karena solusi dijamin ditemukan menurut teorema)

# Algoritma Branch and Bound Dengan Visualisasi

Part 1: Perhitungan Nilai Pembatas

1. Perhitungan 1

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

|  |  |
| --- | --- |
| I | Jumlah Tidak Sesuai |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |
| 5 | 0 |
| 6 | 0 |
| 7 | 0 |
| 8 | 1 (7) |
| 9 | 1 (7) |
| 10 | 1 (7) |
| 11 | 0 |
| 12 | 0 |
| 13 | 1 (12) |
| 14 | 1 (12) |
| 15 | 1 (12) |
| 16 | 9 (8,9,10,7,11,13,14,15,12) |
| Total | 15 |

1. Perhitungan 2

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

Daerah berwarna Abu 🡪 Perhitungan 2 bernilai 1

Daerah berwarna Putih 🡪 Perhitungan 2 bernilai 0

1. Perhitungan 1 + 2

Dari Perhitungan 1 didapat total = 15

Dari perhitungan 2 didapat nilai = 1

Jumlah = 16

Karena jumlah genap, maka solusi dapat dicari

Part 2: Branch and Bound

1. Ambil node pertama dari priority queue

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Queue | Nilai | 5 | 5 | 6 | 7 | 7 | ... | ... |
| Node ke | 1 | 2 | 5 | 3 | 4 | ... | ... |

Pada contoh diatas, maka diambil node ke-1 sehingga:

Node yang dipilih = Node ke-1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Queue | Nilai | 5 | 6 | 7 | 7 | ... | ... | ... |
| Node ke | 2 | 5 | 3 | 4 | ... | ... | ... |

1. Bentuk Pergerakan yang Memungkinkan & Hitung nilainya

Misal Node ke-1 merupakan hasil dari command DOWN dari node akar memiliki status sebagai berikut:

**NODE 1**

Atas : False (bernilai false karena Node 1 dihasilkan dari command DOWN)

Bawah : True

Kiri : True

Kanan : True

Parent : Node 0 (akar)

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

Untuk Node 1, akan digenerate untuk Command DOWN, LEFT, RIGHT sebagai berikut

**NODE 6 (Command DOWN dari NODE 1)**

Atas : False

Bawah : True

Kiri : True

Kanan : True

Parent : Node 1

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

Nilai = 2 (Depth) + 3 (Salah Lokasi) = 5

**NODE 7 (Command LEFT dari NODE 1)**

Atas : True

Bawah : True

Kiri : True

Kanan : False

Parent : Node 1

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

Nilai = 2 + 5 = 7

**NODE 8 (Command RIGHT dari NODE 1)**

Atas : True

Bawah : True

Kiri : False

Kanan : True

Parent : Node 1

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

Nilai = 2 + 5 = 7

Sehingga Setelah Node 1 di cek, maka priority queue menjadi:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Queue | Nilai | 5 | 5 | 6 | 7 | 7 | 7 | 7 |
| Node ke | 2 | 6 | 5 | 3 | 4 | 7 | 8 |

# Source Code

## PriotizedItem.py

Kode untuk membuat item sebuah priority Queue

from dataclasses import dataclass, field

from typing import Any

@dataclass(order=True)

class PrioritizedItem:

    priority: int

    item: Any=field(compare=False)

## Gui.py

Program yang dijalankan untuk menampilkan GUI

import tkinter as tk

from tkinter import messagebox

from fifteen\_puzzle import \*

import time

class Fifteen\_Puzzle:

    def \_\_init\_\_(self):

        for i in range(4):

            for j in range(4):

                entry = tk.Entry(window, width = 5)

                entry.grid(row = i, column = j)

                entry = entry.insert(tk.END, "")

    def show\_each\_move(self, puzzle, iterasi\_ke):

        for i in range(4):

            for j in range(4):

                entry = tk.Entry(window, width = 5)

                entry.grid(row = i, column = j)

                if (puzzle.list[4\*i + j] == 16):

                    entry.insert(tk.END, "")

                else:

                    entry.insert(tk.END, puzzle.list[4\*i+j])

        if(not puzzle.up):

            move = "Down"

        elif(not puzzle.down):

             move = "Up"

        elif(not puzzle.left):

            move = "Right"

        elif(not puzzle.right):

            move = "Left"

        else:

            move = "-"

        text = "Langkah ke-" + str(iterasi\_ke+1) + ": " + move

        tk.Label(window, text = text).grid(row = mod(iterasi\_ke,6), column = 6)

    def showall(self, array\_fifteen\_puzzle):

        global window

        for i in range(len(array\_fifteen\_puzzle)-1,0,-1):

            self.show\_each\_move(array\_fifteen\_puzzle[i],len(array\_fifteen\_puzzle)-i-1)

            time.sleep(1)

            window.update()

def SolveOnClick():

    global array\_fifteen\_puzzle

    global layouts

    filepath =  file\_entry.get()

    array = openFile(filepath)

    hasil = solve(array)

    if(len(hasil)==0):

        tk.messagebox.showinfo("Error","Puzzle Tidak Bisa Diselesaikan!")

    else:

        layout.showall(hasil)

window = tk.Tk()

window.geometry("300x150")

window.title("15-Puzzle Solver")

layout = Fifteen\_Puzzle()

file\_entry = tk.Entry(window, text = "Enter file name")

file\_entry.place(x = 0, y = 85)

solve\_button = tk.Button(window, text = "Solve", width = 15, command = SolveOnClick)

solve\_button.place(x = 5, y = 110)

window.mainloop()

## fifteen\_puzzle.py

Program yang akan mengitung dan menyelesaikan fifteen puzzle tersebut

from operator import mod

from queue import PriorityQueue

from PriotizedItem import PrioritizedItem

import time

import os

class fifteen\_puzzle:

    def \_\_init\_\_(self, list, up, down, left, right):

        self.list = list

        self.up = up

        self.down = down

        self.left = left

        self.right = right

        self.lokasi\_kosong = 0

        self.nilai = 0

        self.kedalaman = 0

        self.id\_parent = 0

    #Hitung total i yang lokasinya salah

    def hitungan\_awal(self):

        hitunngan\_awal = 0

        #iterasi untuk mencari posisi yang salah dari nilai 1 - 16

        for i in range (16):

            #cari posisi nilai ke-i di list

            for j in range (16):

                if(self.list[j] == i+1):

                    break

            #hitung banyaknya nilai k yang lebih kecil dari i tapi posisi lebih besar dari i

            for k in range(16-j-1):

                if(self.list[k+j+1] < i+1):

                    hitunngan\_awal += 1

        return hitunngan\_awal

    #Hitung apakah nilai kosong(nilai ke-16) ada di lokasi kosong

    def arsir\_atau\_tidak(self):

        for i in range (16):

            if(self.list[i] == 16):

                break

        i = mod(i,8)

        if(i==1 or i==3 or i==4 or i==6):

            return 1

        return 0

    #Menyatakan apakah puzzle bisa di selesaikan atau tidak

    def solveable(self):

        nilai = self.hitungan\_awal() + self.arsir\_atau\_tidak()

        if(mod(nilai,2)==0):

            return True

        return False

    def salah\_posisi(self):

        salah=0

        for i in range (16):

            if(self.list[i] != i+1):

                salah+=1

            if(self.list[i]==16):

                self.lokasi\_kosong = i

        return salah

    #Menggerakan puzzle

    def move(self,queue,id\_parent, hashmap):

        if(self.salah\_posisi()==0):

            print("Puzzle berhasil diselesaikan!!")

            return False

        if(self.up):

            if(self.lokasi\_kosong>3):

                self.moveup(queue,id\_parent,hashmap)

        if(self.down):

            if(self.lokasi\_kosong<12):

                self.movedown(queue,id\_parent,hashmap)

        if(self.left):

            if(self.lokasi\_kosong%4!=0):

                self.moveleft(queue,id\_parent,hashmap)

        if(self.right):

            if(self.lokasi\_kosong%4!=3):

                self.moveright(queue,id\_parent,hashmap)

        return True

    #Menggerakan puzzle ke atas

    def moveup(self,queue,id\_parent,hashmap):

        list = self.list[:]

        list[self.lokasi\_kosong], list[self.lokasi\_kosong-4] = list[self.lokasi\_kosong-4], list[self.lokasi\_kosong]

        list = fifteen\_puzzle(list, False, True, True, True)

        list.kedalaman=self.kedalaman+1

        cek\_hash, key = list.in\_hashmap(hashmap)

        if(cek\_hash):

            return

        else:

            list.nilai+=list.kedalaman + list.salah\_posisi()

            list.id\_parent = id\_parent

            queue.put(PrioritizedItem(list.nilai, list))

            hashmap[key] = True

    #Menggerakan puzzle ke bawah

    def movedown(self,queue,id\_parent,hashmap):

        list = self.list[:]

        list[self.lokasi\_kosong], list[self.lokasi\_kosong+4] = list[self.lokasi\_kosong+4], list[self.lokasi\_kosong]

        list = fifteen\_puzzle(list, False, True, True, True)

        cek\_hash, key = list.in\_hashmap(hashmap)

        if(cek\_hash):

            return

        else:

            list.kedalaman=self.kedalaman+1

            list.nilai+=list.kedalaman + list.salah\_posisi()

            list.id\_parent = id\_parent

            queue.put(PrioritizedItem(list.nilai, list))

            hashmap[key] = True

    #Menggerakan puzzle ke kiri

    def moveleft(self,queue,id\_parent,hashmap):

        list = self.list[:]

        list[self.lokasi\_kosong], list[self.lokasi\_kosong-1] = list[self.lokasi\_kosong-1], list[self.lokasi\_kosong]

        list = fifteen\_puzzle(list, True, True, True, False)

        cek\_hash, key = list.in\_hashmap(hashmap)

        if(cek\_hash):

            return

        else:

            list.kedalaman=self.kedalaman+1

            list.nilai+=list.kedalaman+list.salah\_posisi()

            list.id\_parent = id\_parent

            queue.put(PrioritizedItem(list.nilai, list))

            hashmap[key] = True

    #Menggerakan puzzle ke kanan

    def moveright(self,queue,id\_parent,hashmap):

        list = self.list[:]

        list[self.lokasi\_kosong], list[self.lokasi\_kosong+1] = list[self.lokasi\_kosong+1], list[self.lokasi\_kosong]

        list = fifteen\_puzzle(list, True, True, False, True)

        cek\_hash, key = list.in\_hashmap(hashmap)

        if(cek\_hash):

            return

        else:

            list.kedalaman=self.kedalaman+1

            list.nilai+=list.kedalaman+list.salah\_posisi()

            list.id\_parent = id\_parent

            queue.put(PrioritizedItem(list.nilai, list))

            hashmap[key] = True

    #Mengecek sate puzzle dalam hash table

    def in\_hashmap(self,hash\_table):

        key=''

        for i in range(16):

            key+=str(self.list[i])+'-'

        if(key in hash\_table):

            return True,"-"

        else:

            return False,key

    #Print Puzzle

    def print\_puzzle(self):

        for i in range (4):

            print("   ",end="")

            for j in range (4):

                if(self.list[i\*4+j] == 16):

                    print("  ", end=" ")

                else:

                    print("%2d" % self.list[i\*4+j], end=" ")

            print("")

#Fungsi Tambahan Untuk GUI

#Baca data dari file

def openFile(nama\_file):

    curr\_dir = os.path.dirname(os.path.abspath(\_\_file\_\_))

    try:

        f = open(curr\_dir+"/../test/"+nama\_file, "r")

    except:

        print("File tidak ditemukan")

        exit()

    array=[]

    for x in f:

        x = x.split(' ')

        array.append((x[0]))

        array.append((x[1]))

        array.append((x[2]))

        array.append((x[3]))

    for i in range(16):

        if array[i]=='-':

            array[i]=16

        else:

            array[i]=int(array[i])

    f.close()

    return array

def solve(array):

    #inisialisasi variabel

    puzzle = fifteen\_puzzle(array,True,True,True,True);

    selesai = False

    id\_dikerjakan = 0

    queue = PriorityQueue()

    urutan\_pengerjaan = []

    hasil\_print = []

    puzzle.salah\_posisi()

    puzzle\_awal = puzzle

    hashmap = {}

    #memasukkan akar ke queue

    start\_time = time.time()

    queue.put(PrioritizedItem(puzzle.nilai, puzzle))

    while(not queue.empty() and not selesai):

        puzzle = queue.get().item

        urutan\_pengerjaan.append(puzzle)

        id\_dikerjakan += 1;

        if(not puzzle.move(queue, id\_dikerjakan,hashmap)):

            stop\_time = time.time()

            break

    #Menampilkan Hasil

    while(not queue.empty()):

        queue.get()

        id\_dikerjakan+=1

    # Mnecari dan Mencetak Path yang menghasilkan tujuan

    hasil\_print.append(puzzle\_awal)

    while(puzzle.id\_parent != 0):

        hasil\_print.append(puzzle)

        puzzle = urutan\_pengerjaan[puzzle.id\_parent-1]

    return hasil\_print

## main.py

Program utama yang dijalankan. Program akan membaca data dari file dan akan memanggil fifteen\_puzzle.py. Seletah selesai, program juga akan menampilkan path menuju solusi

# DISCLAIMER!!!!

# Semua Puzzle yang bernilai Genap dapat diselesaikan dengan

# Heuristik sesuai Spek, Namun tidak semua dapat diselesaikan

# Dengan cukup cepat sehingga jika puzzle kompleks (>30an langkah)

# Maka ada kemungkinan python akan mengalami error timeout sebelum hasil ditemukan

import queue

from fifteen\_puzzle import \*

from queue import PriorityQueue

from PriotizedItem import PrioritizedItem

nama\_file = input("Masukkan nama file yang ada di Folder Test: ")

print("==================")

array = openFile(nama\_file)

#===========================================

#Solve 15 Puzzle

puzzle = fifteen\_puzzle(array, True, True, True, True)

if(not puzzle.solveable()):

    print("Puzzle tidak bisa di selesaikan")

    print("karena puzzle bernilai ganjil yakni: ",puzzle.hitungan\_awal()+puzzle.arsir\_atau\_tidak())

    print("==================")

    puzzle.print\_puzzle()

    print("==================")

else:

    #inisialisasi variabel

    selesai = False

    id\_dikerjakan = 0

    queue = PriorityQueue()

    urutan\_pengerjaan = []

    hasil\_print = []

    puzzle.salah\_posisi()

    puzzle\_awal = puzzle

    hashmap = {}

    #memasukkan akar ke queue

    print("Searching.......")

    start\_time = time.time()

    queue.put(PrioritizedItem(puzzle.nilai, puzzle))

    while(not queue.empty() and not selesai):

        puzzle = queue.get().item

        urutan\_pengerjaan.append(puzzle)

        id\_dikerjakan += 1;

        if(not puzzle.move(queue, id\_dikerjakan,hashmap)):

            stop\_time = time.time()

            break

    #Menampilkan Hasil

    print("Total Pengecekan: ",id\_dikerjakan-1)

    while(not queue.empty()):

        queue.get()

        id\_dikerjakan+=1

    print("Total Pembangkitan: ",id\_dikerjakan-1)

    # Mnecari dan Mencetak Path yang menghasilkan tujuan

    while(puzzle.id\_parent != 0):

        hasil\_print.append(puzzle)

        puzzle = urutan\_pengerjaan[puzzle.id\_parent-1]

    hasil\_print.append(puzzle\_awal)

    print("Puzzle diselesaikan dalam:",end=" ")

    print(len(hasil\_print)-1,"langkah")

    print("Puzzle diselesaikan dalam:",end=" ")

    print("%.10s detik" % (stop\_time - start\_time))

    print("==================")

    for i in range (len(hasil\_print)-1,-1,-1):

        hasil\_print[i].print\_puzzle()

        print("Command: ",end="")

        if(not hasil\_print[i-1].up):

            print("Down")

        elif(not hasil\_print[i-1].down):

            print("Up")

        elif(not hasil\_print[i-1].left):

            print("Right")

        elif(not hasil\_print[i-1].right):

            print("Left")

        else:

            print("-")

        print("==================")

# Hasil Pengujian

## Berhasil1.txt

A screen with numbers and letters on it

Description automatically generated with low confidence

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## Berhasil2.txt

A picture containing text, electronics

Description automatically generated

A picture containing calendar

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## Berhasil3.txt

A picture containing text, electronics

Description automatically generated

Text

Description automatically generated

...

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated

## Gagal1.txt

A picture containing text, electronics

Description automatically generated

Text

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

## Gagal2.txt

A picture containing text, electronics, close

Description automatically generated

Text

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

# Lampiran

Table

Description automatically generated

√

√

√

√

√

Link Drive:

<https://drive.google.com/drive/folders/1C_vXKNOrep4xt0j0xMcm7B-HoZUgxGih?usp=sharing>