Algoritma Breadth-First Search di python

from collections import deque

from state import State

goal\_state = [ # State yang dituju

1, 2, 3,

4, 5, 6,

7, 8, 0

]

initial\_state = [ # State awal

0, 8, 7,

6, 5, 4,

3, 2, 1

]

queue = deque([State(initial\_state, None, 0, goal\_state)]) # masukkan state

# awal ke queue

queue\_popped = [] # digunakan untuk menyimpan solusi

queue\_res = deque([]) # digunakan untuk menyimpan solusi yg sudah di-backtrace

end\_state = None # menyimpan State langkah akhir

maxDepth = 0

while len(queue):

now = queue.popleft() # pop queue paling depan

isSameBefore = False

for state in queue\_popped: # mencek jika langkah sudah pernah dilakukan

if now.isSame(state.board):

isSameBefore = True # jika belum dilakukan tandai

break

if isSameBefore:

continue # jika sudah, abaikan

queue\_popped.append(now) # masukkan ke queue solusi

upState = now.expandUp() # lakukan langkah ke atas, ke bawah

downState = now.expandDown() # ke kiri dan ke kanan

leftState = now.expandLeft()

rightState = now.expandRight()

if upState is not None: # Jika langkah dapat dilakukan

if upState.isSame(goal\_state): # Jika langkah yang diambil = tujuan

end\_state = upstate # simpan langkah sebagai langkah akhir

break

if upState.cost > maxDepth:

maxDepth = upState.cost

queue.append(upState) # masukkan ke queue

if downState is not None:

if downState.isSame(goal\_state):

end\_state = downState

break

if downState.cost > maxDepth:

maxDepth = downState.cost

queue.append(downState)

if rightState is not None:

if rightState.isSame(goal\_state):

end\_state = rightState

break

if rightState.cost > maxDepth:

maxDepth = rightState.cost

queue.append(rightState)

if leftState is not None:

if leftState.isSame(goal\_state):

end\_state = leftState

break

if leftState.cost > maxDepth:

maxDepth = leftState.cost

queue.append(leftState)

# solusi yang didapatkan akan terbalik

while end\_state is not None: # maka lakukan backtracking parent untuk

# membalikkan queue

queue\_res.appendleft(end\_state)

end\_state = end\_state.parent