

200302211

Kecerdasan Buatan *(Artificial Intelligence)*

Program Studi Informatika

***Session 3: Uninformed and
Informed Search***

Lecturer: Ega Dioni Putri, S. T. , M. M. G.



Kilas Balik Materi Sesi 1 dan 2

Session 1 and 2 Review

- ★ Introduction
- ★ Lecture T&C
- ★ AI Definition
- ★ AI Journey
- ★ Type of AI
- ★ Application of AI in nowadays
- ★ Quiz, Vicon, Module distribution

Session 1

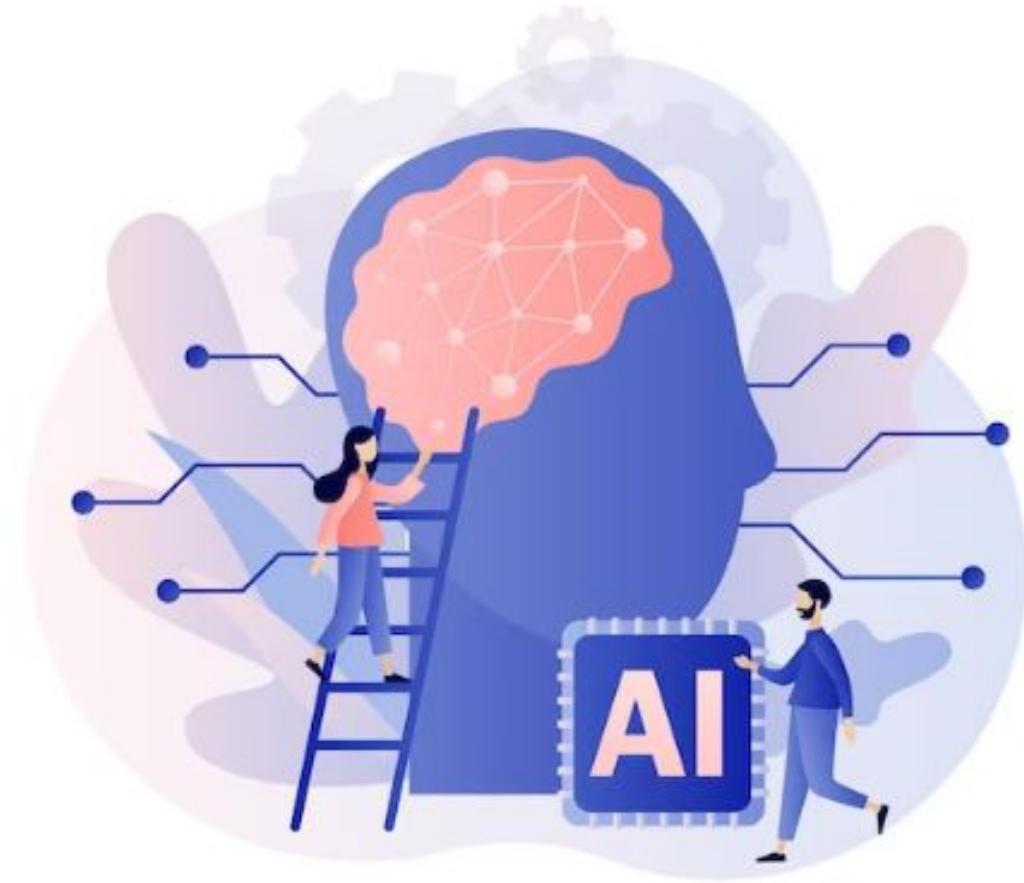
- ★ Lecture Goal
- ★ Definition of “Problem” in AI
- ★ Problem as Input
- ★ Problem as State Space
- ★ Problem Representation
- ★ Problem Solving
- ★ Quiz, Module distribution

Session 2

Session 3 - Uninformed and Informed Search

Part #1

Konsep Pencarian (*The Concept of Search*)



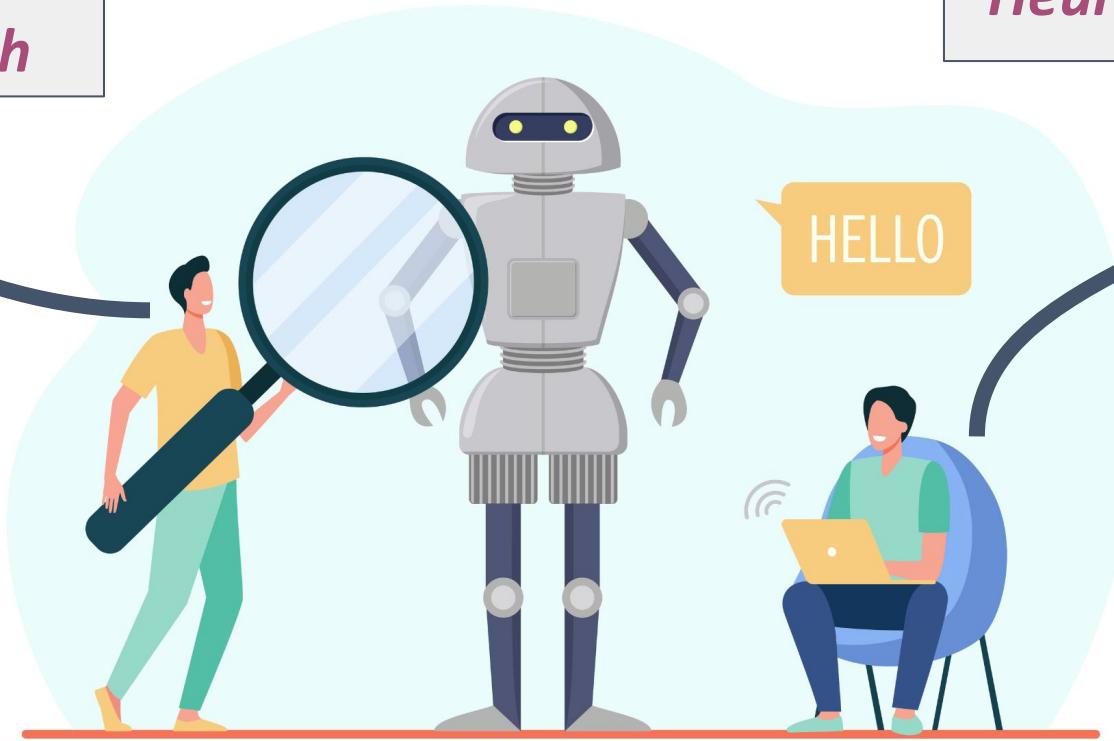


Teknik Pencarian dalam Kecerdasan Buatan

Searching in AI

Pencarian Buta
*Blind search /
Uninformed search*

*"Look around, don't
know where to find
the right answer"*



Pencarian Terbimbing
Heuristic / Informed Search

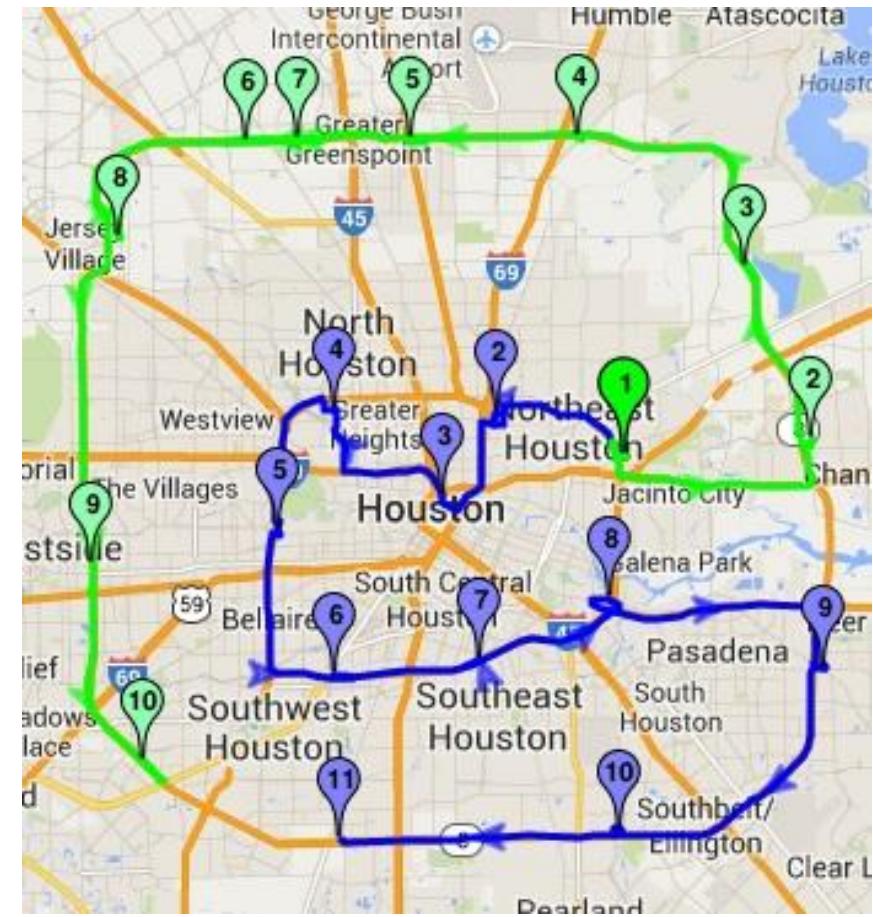
*"Know some
information that
sometimes helpful!"*



Imagine A Route Planning in A Map

An Easy Way to Understand “Search” in AI

- Real world application to search tree:
Cities are nodes, roads are links
World (set of the cities) is state space
- World characteristics:
 - **finite and enumerable** (terbatas, dapat dihitung)
 - **deterministic** (satu rute, satu start - goal yang jelas)
- Problem examples: distance, total time





Definisi Pencarian

Formal Definition of “Search” in AI

Contoh penggambaran masalah pada pencarian

- **Set of states** (ruang keadaan): S
- **Initial state** (kondisi awal)
- **Goal state** (tujuan/solusi): $S \rightarrow \{s,a\}$
mengembalikan *reached state* saat aksi a dilakukan pada state s
- **Operators/actions**: $S \rightarrow s$
- **Rules**:
 - Path cost: (S,O)
 - Sum of cost: $S c(S,O)$



Parameter Kualitas Pencarian

Criteria for Algorithms in Searching

Baik tidaknya suatu algoritma pencarian dapat dilihat dari kriteria berikut:

- ***Completeness*** → Apakah metode tersebut menjamin penemuan solusi jika solusinya memang ada?
- ***Time complexity*** → Berapa lama waktu yang diperlukan?
- ***Space complexity*** → Berapa banyak memori yang diperlukan?
- ***Optimality / Solution Quality*** → Apakah metode tersebut menjamin menemukan solusi yang terbaik jika terdapat beberapa solusi yang berbeda?

Session 3 - Uninformed and Informed Search

Part #2

Uninformed Search





Apa Itu Pencarian Buta?

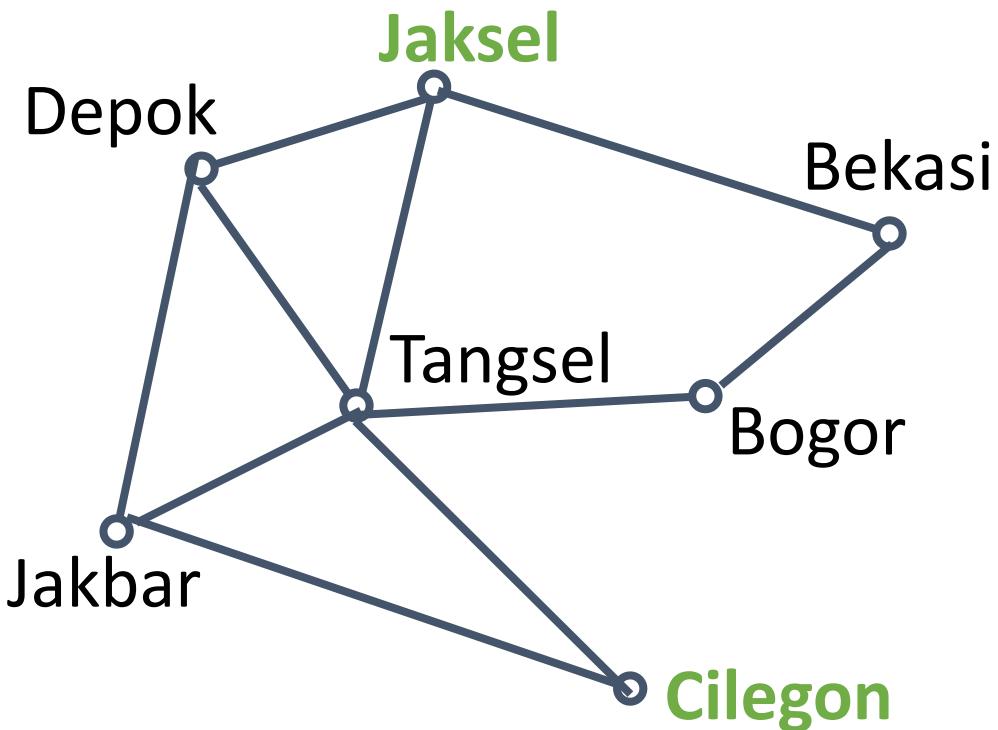
What is uninformed search / blind search?

- Pencarian yang mengandalkan pengecekan semua kemungkinan solusi, tidak ada informasi awal tentang domain
 - Tidak ada petunjuk jawaban yang benar ke arah mana (buta)
 - Tidak tahu apakah hasil yang ditemukan adalah solusi terbaik atau bukan
-
- *A search that relies on checking all possible solutions, no information about its domain*
 - *No clue as to which way is the correct answer (blind)*
 - *No idea whether the result found is the best solution or not*



Apa Itu Pencarian Buta? (2)

What is uninformed search / blind search? (2)



From Jaksel to Cilegon: We don't know which node to explore first. Is it Depok, Tangsel, or Bekasi?

Pada pencarian buta, misalkan dicari rute dari Jaksel ke Cilegon, kita tidak punya petunjuk titik mana dulu yang harus dijelajahi pada level pertama untuk mendapatkan jarak terdekat:

Depok?

Tangsel?

atau Bekasi?



Algoritma pada Teknik Pencarian Buta

Blind Search Algorithms

Breadth-First Search (BFS)	cari per level dulu, baru next level
Depth-First Search (DFS)	selesaikan pencarian di satu jalur dulu
Depth-Limited Search (DLS)	membatasi DFS dengan level limit
Uniform Cost Search (UCS)	mencari berdasarkan biaya terendah
Iterative-Deepening Search (IDS)	kolaborasi cara BFS dan DFS
Bi-Directional Search (BDS)	mencari dari start ke goal dan vice versa



Breadth First Search

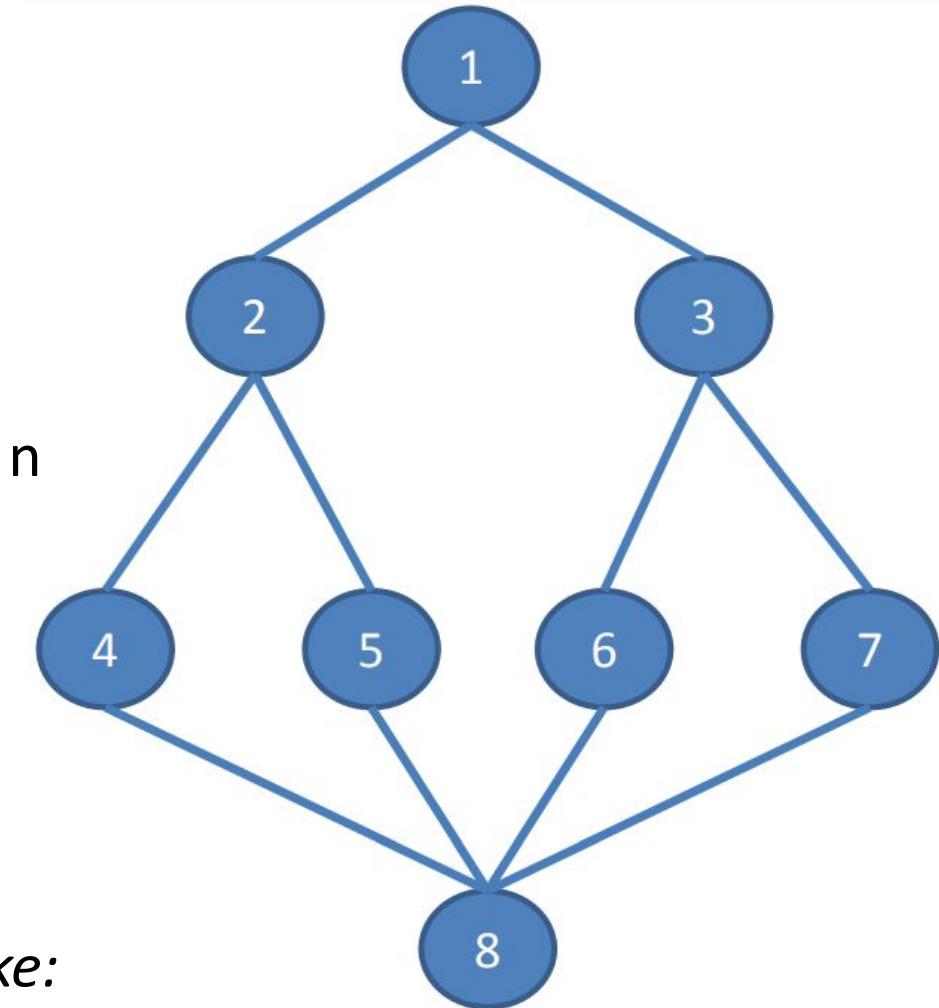
Pencarian Melebar

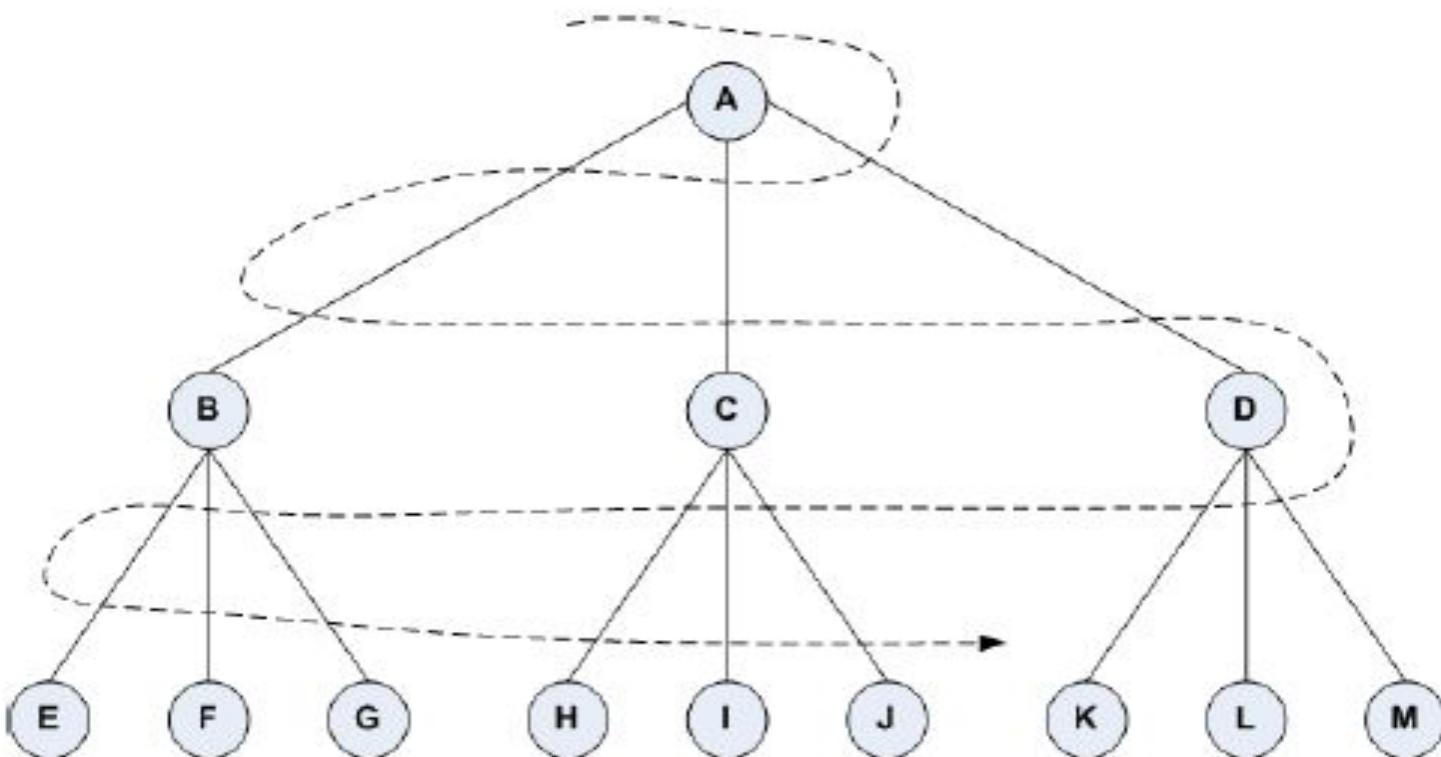
Algoritma

1. Pencarian dimulai dari simpul akar, misal di level n
2. Kunjungi simpul di level berikutnya ($n+1$), lalu yang semua simpul bertetangga dengan simpul tersebut
3. Lanjutkan dengan simpul di level atasnya ($n+2$) dan semua simpul yang bertetangga, sampai ditemukan solusi

Searching order be like:

$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8$





BFS Algorithm

1. *Search starts from the root node, for example at level n*
2. *Visit the node at the next level ($n+1$), then all the nodes that are neighbors of that node*
3. *Continue with the node at the level above ($n+2$) and all the neighboring nodes, until a solution is found.*

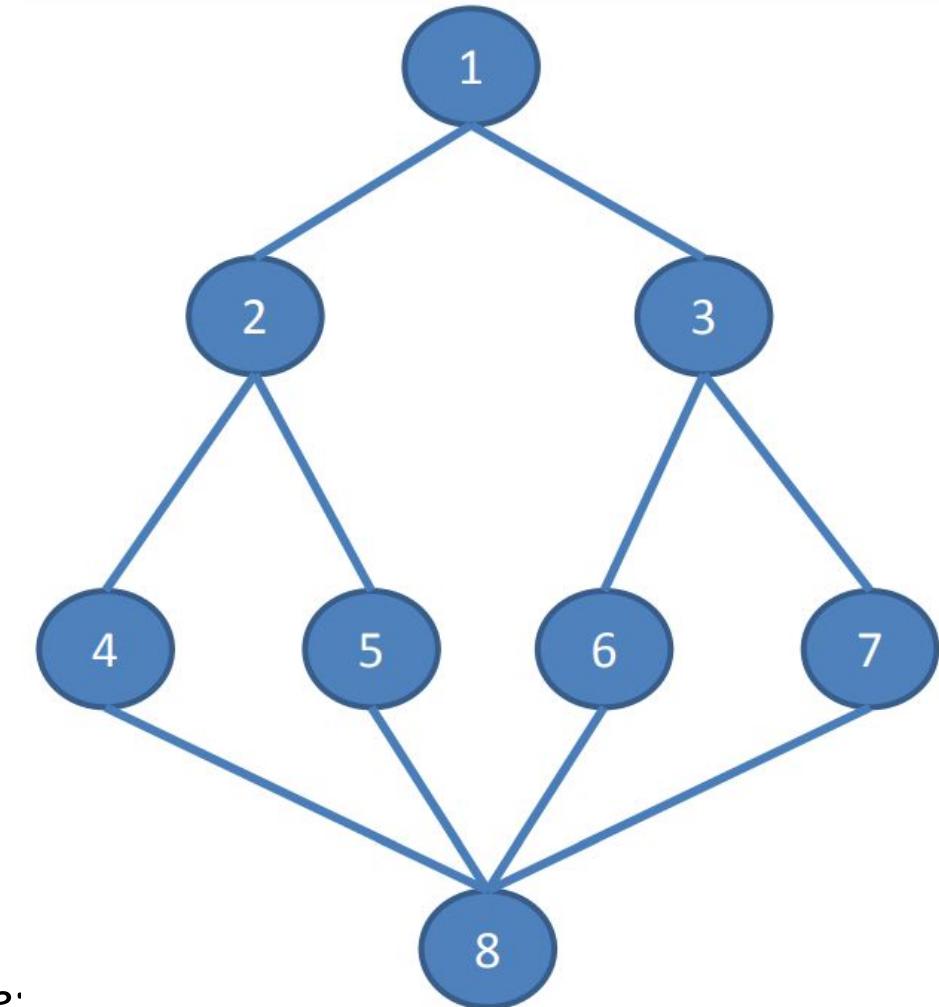


Depth First Search

Pencarian Mendalam

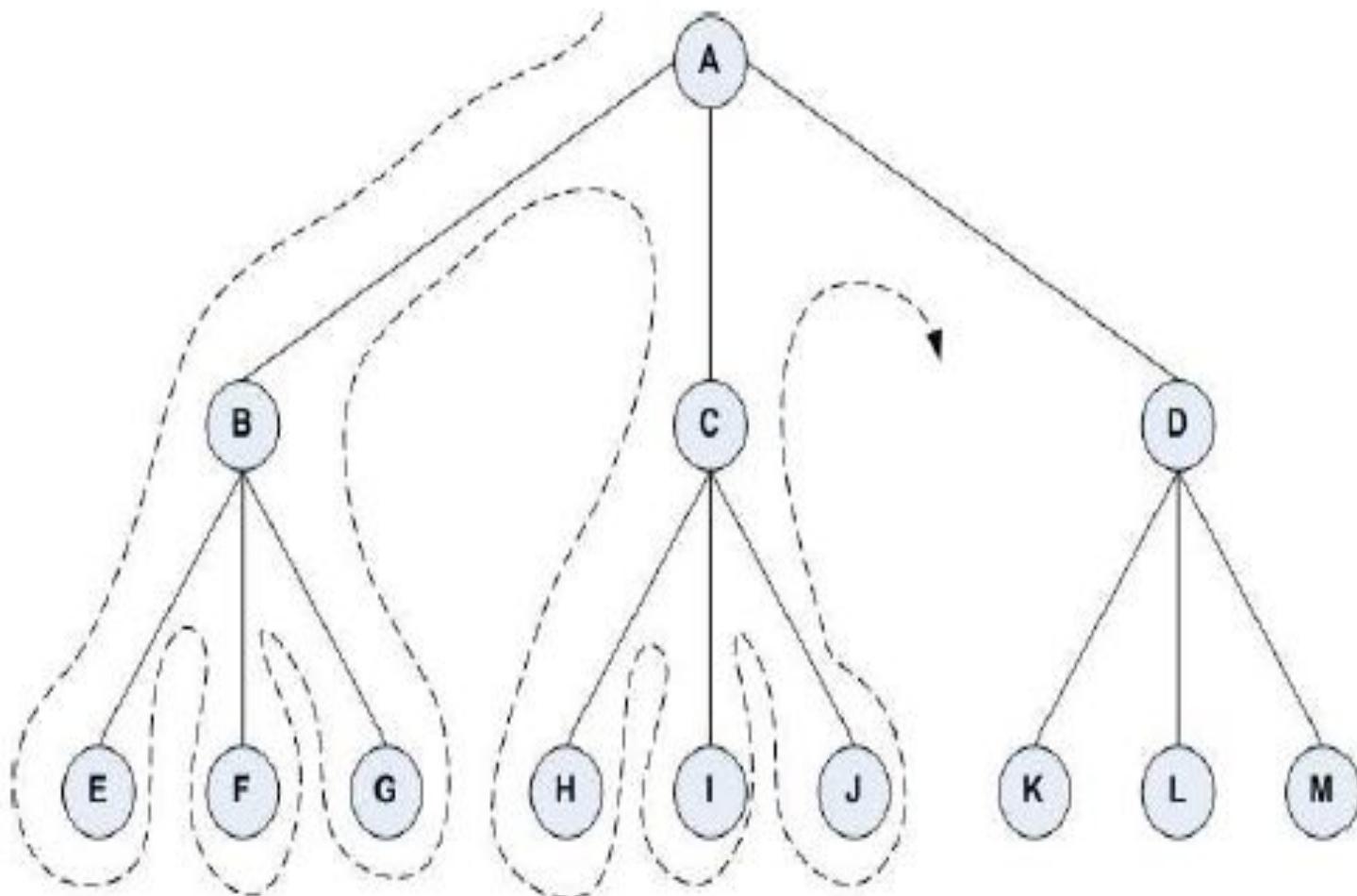
Algoritma

1. Pencarian dimulai dari simpul akar, misal di level n
2. Kunjungi simpul di level berikutnya ($n+1$), lalu simpul anak dari simpul tersebut, hingga selesai menjelajahi semua simpul di satu cabang
3. Runut balik ke simpul pada cabang tetangga yang selevel dengan simpul terakhir yang dikunjungi, lalu jelajahi simpul-simpul ibunya yang belum dikunjungi di cabang tersebut
4. Ulangi langkah 2-3 pada cabang lain hingga seluruh simpul ditelusuri



Searching order be like:

$1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 5 \rightarrow 6 \rightarrow 3 \rightarrow 7$



DFS Algorithm

1. *Search starts from the root node, say at level n*
2. *Visit the node at the next level ($n+1$), then the child nodes of that node, until you have explored all the nodes in one branch*
3. *Backtrack to the node on its neighboring branch that is at the same level as the last node visited, then explore its unvisited mother nodes.*
4. *Repeat steps 2-3 on other branches until all the nodes are explored*



BFS vs DFS

Breadth-first Search (BFS)

Uses queue data structure to store the nodes to be visited

Traverses a graph level-wise

Not suitable for decision-making trees used in games or puzzle

More suitable when the target node is closer to the source

Depth-first Search (DFS)

Uses stack to store the nodes to be visited

Traverses a graph depth-wise

More suitable for game or puzzle problems

More suitable when the target node is away from the source node



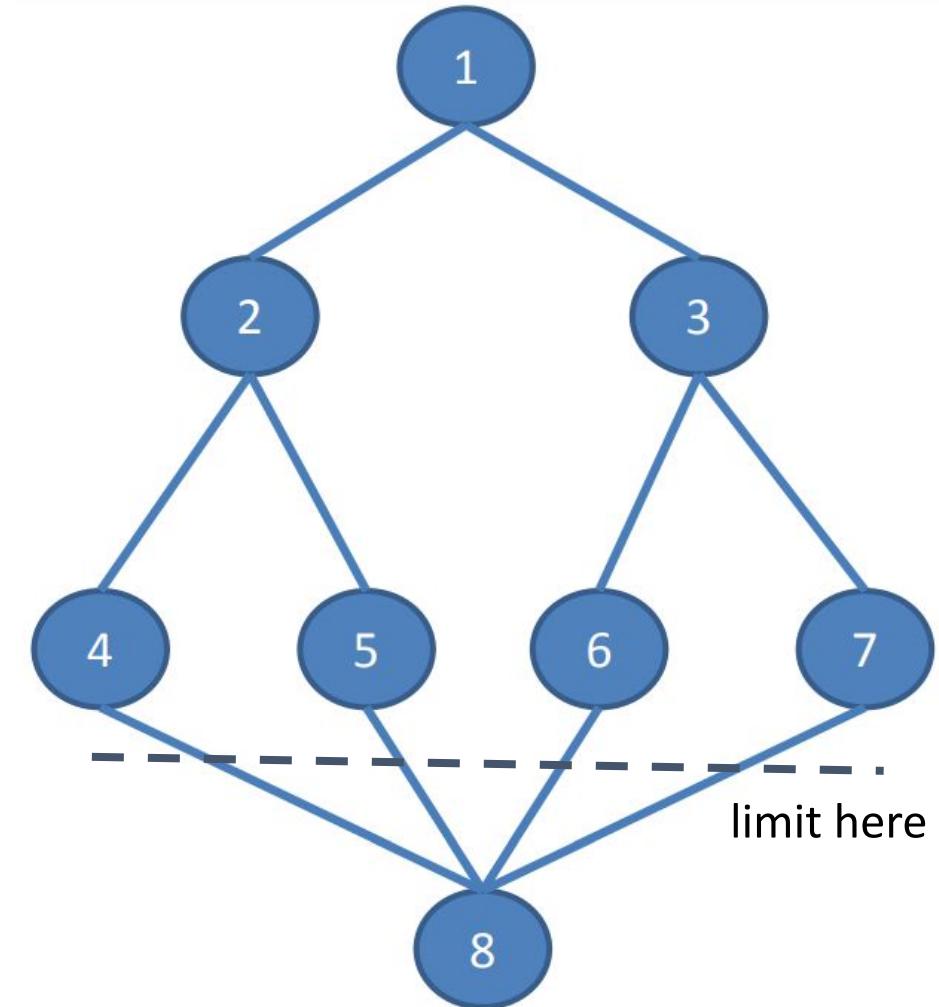
Depth-Limited Search

Pencarian Mendalam Terbatas

Algoritma

1. Pencarian dimulai dari simpul akar, misal di level n
2. Kunjungi simpul di level berikutnya ($n+1$), lalu simpul anak dari simpul tersebut, hingga selesai menjelajahi simpul di level terakhir batas kedalaman
3. Runut balik ke simpul pada cabang tetangga yang selevel dengan simpul terakhir yang dikunjungi, lalu jelajahi simpul-simpul ibunya yang belum dikunjungi di cabang tersebut
4. Ulangi langkah 2-3 pada cabang lain hingga seluruh simpul ditelusuri tanpa mengecek simpul di level yang melebihi batas

Searching order be like: 1 → 2 → 4 → 5 → 6 → 3 → 7



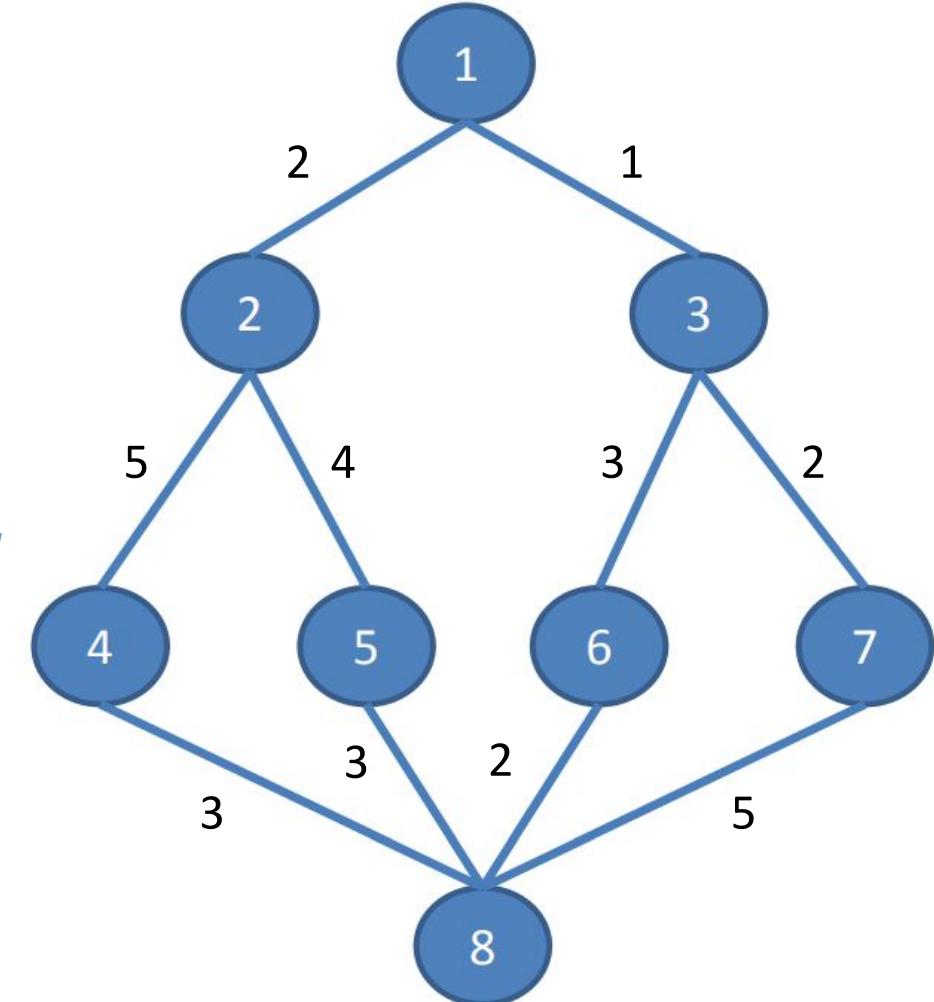
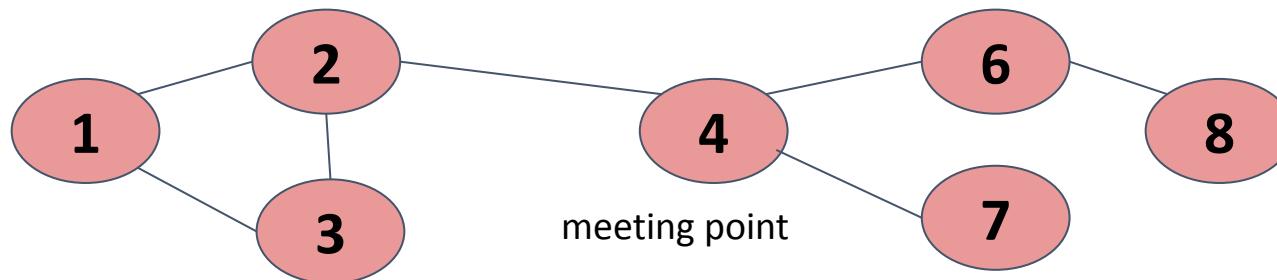


Mini task!

*How about the steps in
Uniform Cost Search?
Iterative-Deepening Search?
Bi-Directional Cost Search?*

Write down the order of visited nodes as well!

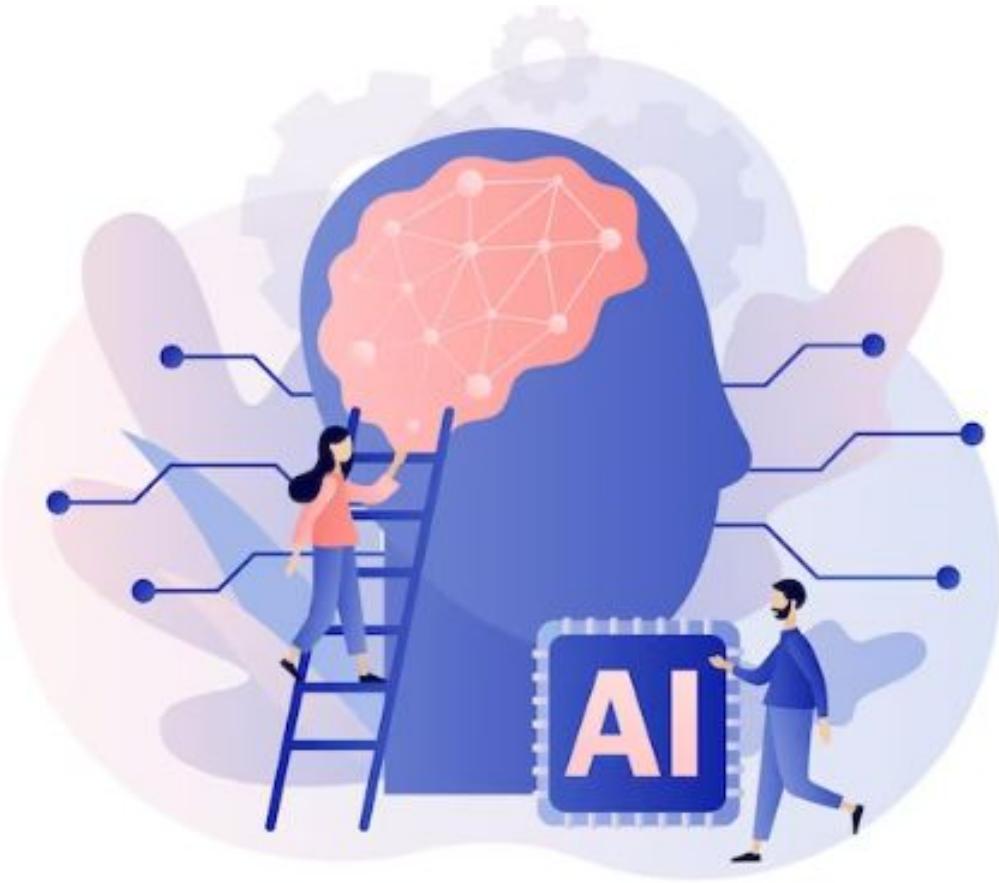
graph for bi-directional search:



Session 3 - Uninformed and Informed Search

Part #3

Informed Search





Pencarian Terbimbing/Terinformasi = Heuristik

Informed Search = Heuristic Search

Oxford Dictionary

Heuristic:

1. *enabling someone to discover or learn something for themselves.*
2. *in Comp. term: proceeding to a solution by trial and error or by rules that are only loosely defined.*

Kamus Besar Bahasa Indonesia

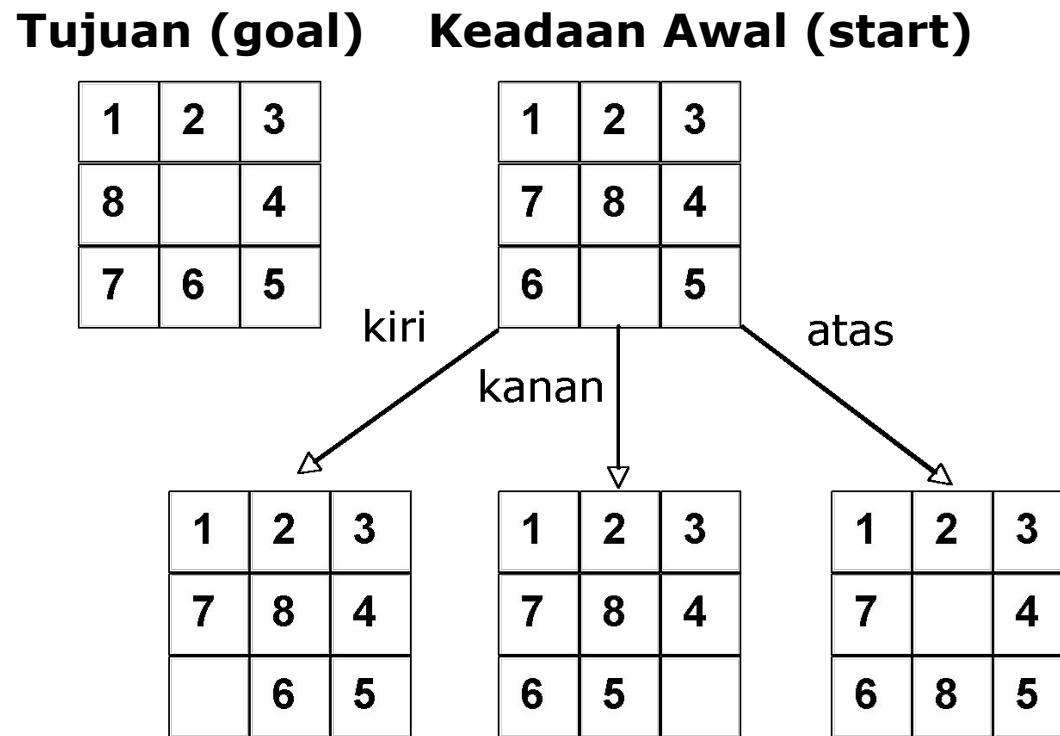
Heuristik:

1. berkaitan dengan formulasi yang biasanya **spekulatif**, berfungsi sebagai panduan dalam penyelidikan atau pemecahan masalah
2. berkaitan dengan metode pendidikan yang pembelajarannya berlangsung melalui penemuan berdasarkan pencarian oleh siswa atau **pengalaman** siswa sendiri
3. kajian dan penerapan metode atau prosedur analitis yang dimulai dengan **perkiraan yang tepat** dan mengecek ulang sebelum memberi kepastian



Pencarian Heuristik dalam Permainan 8 Puzzle(1)

Heuristic Search in 8-Puzzle Game (1)



- Operator yang bisa diterapkan:
 1. Ubin kosong geser ke kanan
 2. Ubin kosong geser ke kiri
 3. Ubin kosong geser ke atas
 4. Ubin kosong geser ke bawah
- Pada langkah awal, hanya 3 operator yang bisa digunakan (1, 2, 3)
- Jika menggunakan pencarian buta, tidak perlu mengetahui operasi apa yang akan dikerjakan (sembarang)
- Dalam pencarian heuristik, perlu diberikan informasi khusus dalam **domain** tersebut



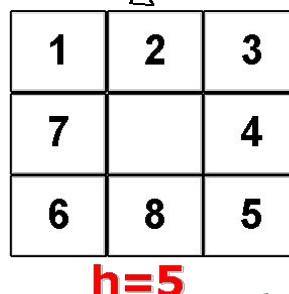
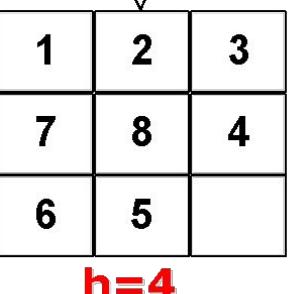
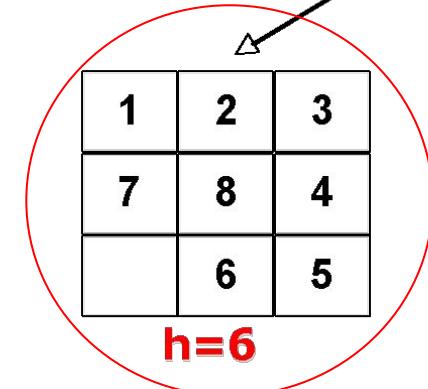
Pencarian Heuristik dalam Permainan 8 Puzzle (2)

Heuristic Search in 8-Puzzle Game (2)

Tujuan (goal) Keadaan Awal (start)

1	2	3
8		4
7	6	5

1	2	3
7	8	4
6		5



kiri

kanan

atas

+ ditambah
informasi

- Untuk jumlah ubin yang menempati posisi benar (sesuai kondisi tujuan), nilai heuristik yang lebih tinggi adalah yang lebih diharapkan (lebih baik) → Simpul paling kiri paling tinggi nilainya



Pencarian Heuristik dalam Permainan 8 Puzzle (3)

Heuristic Search in 8-Puzzle Game (3)

Tujuan (goal)

1	2	3
8		4
7	6	5

Keadaan Awal (start)

1	2	3
7	8	4
6		5

kiri

atas

1	2	3
7	8	4
6	5	

h=4

1	2	3
7		4
6	8	5

h=3

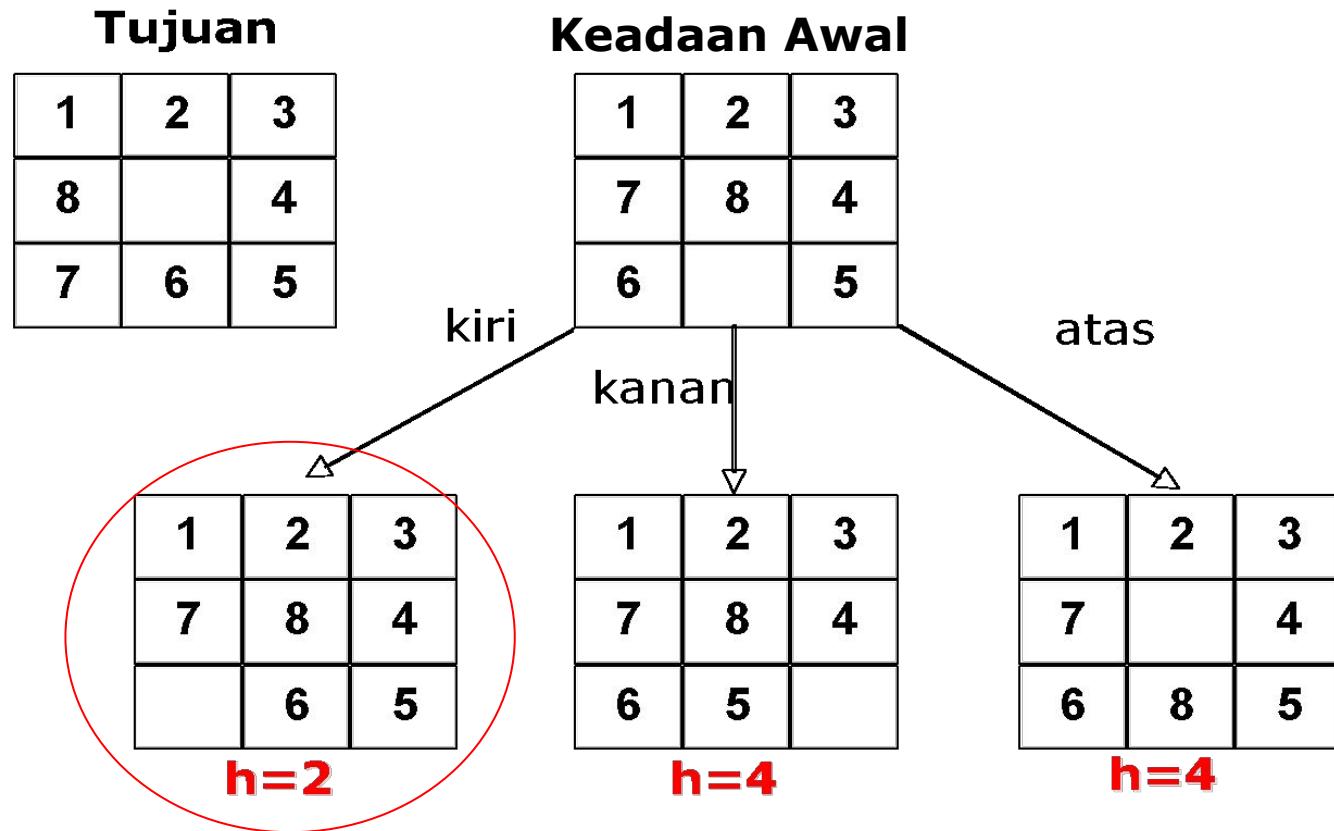
h=2

- Untuk jumlah ubin yang menempati posisi salah, nilai heuristik yang lebih rendah adalah yang diharapkan (lebih baik) → Simpul paling kiri paling rendah nilainya, artinya paling sedikit salahnya



Pencarian Heuristik dalam Permainan 8 Puzzle (4)

Heuristic Search in 8-Puzzle Game (4)



- Dalam menghitung total gerakan yang diperlukan untuk mencapai tujuan, jumlah yang lebih kecil adalah yang diharapkan (lebih baik) → Lebih sedikit langkah, lebih hemat biayanya

English explanation

Heuristic Search in 8-Puzzle Game (1)

- ❑ Operators that can be applied:
 - ❑ Empty tile shift right
 - ❑ Empty tile shift left
 - ❑ Empty tile shift up
 - ❑ Empty tile shift down
- ❑ In the initial step, only 3 operators can be used (1, 2, 3)
- ❑ If using b* search, there is no need to know what operation will be performed (arbitrary)
- ❑ In heuristic search, it is necessary to provide specific information in the domain

English explanation

Heuristic Search in 8-Puzzle Game (2-4)

- ❑ For the number of tiles that has set the correct positions (according to the goal state), a higher heuristic value is more expected (better) → In the picture, shown that the leftmost node has the highest value
- ❑ For a given number of tiles in the wrong position, a lower heuristic value is expected (better) → In the picture, shown that the leftmost node has the lowest value, meaning it is the least wrong
- ❑ In calculating the total number of moves required to reach a goal, the smaller the number the more expected (better) → Fewer steps, more cost-effective



A* Search

Pencarian A*

- Memperhitungkan apakah biaya untuk mencapai simpul saat ini dan estimasi biaya untuk mencapai tujuan TIDAK LEBIH dari biaya aktual
- Menyimpan riwayat simpul-simpul yang sudah dikunjungi sehingga tidak terjadi *revisit*
- Menampung informasi tentang simpul-simpul yang hendak diperlakukan, tetapi hanya memilih simpul yang paling optimal

nilai g = “biaya” dari simpul awal ke saat ini (jumlah simpul yang telah diperlakukan)

nilai h = “biaya” estimasi ke simpul tujuan (jumlah puzzle yang masih salah)

nilai f = $g + h$ (dipilih nilai yang paling kecil sebagai ukuran “optimal”)

biaya aktual = 5 (dihitung dari perbedaan keadaan awal dan tujuan)

Tujuan (goal)

1	2	3
8		4
7	6	5

Keadaan Awal (start)

1	2	3
7	8	4
6		5

$$g=0, h=3$$
$$f=0+3=3$$

atas

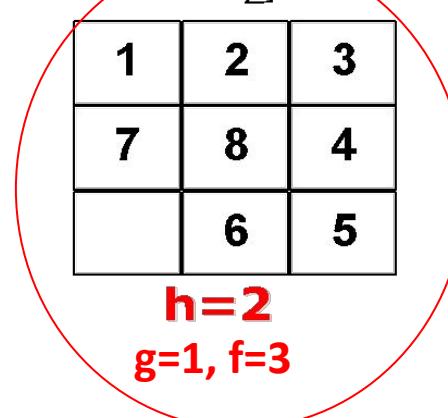
1	2	3
7	8	4
6	5	

1	2	3
7		4
6	8	5

1	2	3
7		4
6	8	5

$$h=4$$
$$g=1, f=5$$

$$h=4$$
$$g=1, f=5$$



English explanation

A* Search Example

A* characteristics

- Calculate and make sure that *the cost to reach the current node from the start state PLUS the estimated cost to reach the goal* is NO MORE than the actual cost
- Save the list of visited nodes for avoiding repetition
- Keep the list of nodes to be explored before selecting the most optimal node

actual cost = 5

calculated from the start state and goal difference

A* components:

- **Heuristics $h(n)$**
The estimated “cost” to the goal node (in the picture, it shows the number of unset puzzles)
- **Heuristic function $g(n)$**
The “cost” from the starting node to the current node (in the picture, it shows the number of nodes explored)
- **Evaluation function $f(n)$**
 $f(n) = g(n) + h(n)$
Selected the least value to get an optimal solution, it should not exceed the *actual cost*



Keadaan Awal (start)

1	2	3
7	8	4
6		5

Tujuan (goal)

1	2	3
8		4
7	6	5



*How about the steps in
Greedy Best-First Search?
Draw the search tree and show the goal node*

Mini task!

TERIMA KASIH

Sampai jumpa di sesi berikutnya!



Jangan lupa cek LMS untuk kuis dan berikan umpan balik ☺