

# Informed Search (Heuristic) & Eksplorasinya

Chastine Fatichah  
Departemen Teknik Informatika  
Maret 2023



IF

# Capaian Pembelajaran Matakuliah

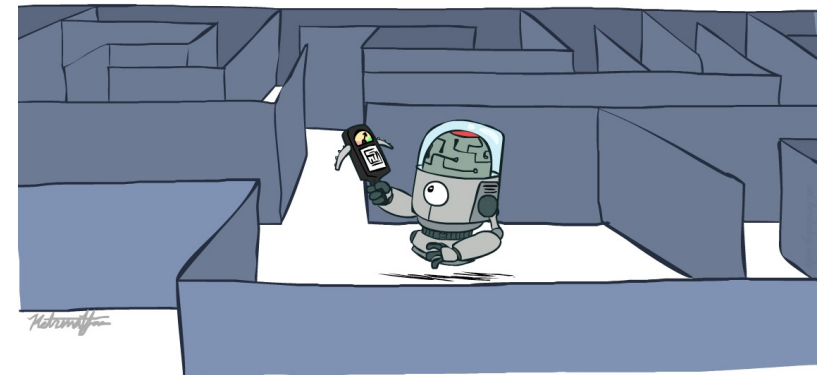
Mahasiswa mampu menjelaskan, mengidentifikasi, merancang, dan menerapkan *intelligent agent* untuk *problem* yang sesuai dengan memanfaatkan algoritma pencarian yang meliputi *uninformed search*, *informed search*, *heuristic search*, *adversarial search*, serta algoritma *search* untuk *Constraint Satisfaction Problem*

## *Informed search strategies*

- *Heuristic*
- *Greedy Best-First Search*
- *A\* Search*
- *Iterated Deepening A\* (IDA)*
- *Recursive Best-first Search (RBFS)*
- *Simplified Memory Bounded A\* (SMA)*

# Informed Search

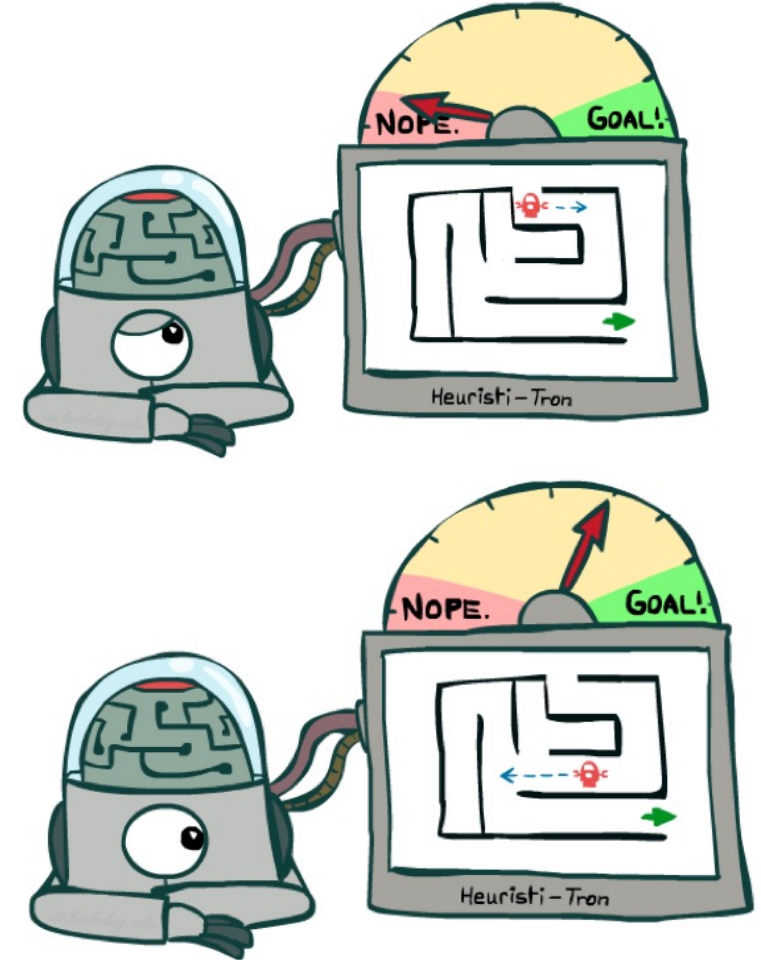
- **Uninformed Search**: menggenerate state baru, di cek apakah *goal* atau tidak → kurang efisien
  - Uniform Cost Search (UCS)
    - Strategi: *expand lowest path cost*
    - Kelebihan: *complete and optimal*
    - Kekurangan: melakukan eksplorasi di semua arah dan tidak ada informasi tentang lokasi *goal*
- **Informed Search**: menggunakan informasi tambahan → lebih efisien
  - *Heuristic function* → informasi estimasi menuju goal
  - Best-First Search
    - *Greedy Best-First*
    - $A^*$
    - *Iterated Deepening  $A^*$  (IDA)*
    - *Recursive Best-first Search (RBFS)*
    - *Simplified Memory Bounded  $A^*$  (SMA)*





# Heuristics Function

- **Fungsi Heuristik**
  - Sebuah fungsi yang mengestimasi seberapa dekat *state* sekarang ke *state* tujuan (*goal*)
  - Dirancang untuk *search problem* tipe tertentu
  - Contoh: *Manhattan distance*, *Euclidean distance*, ...



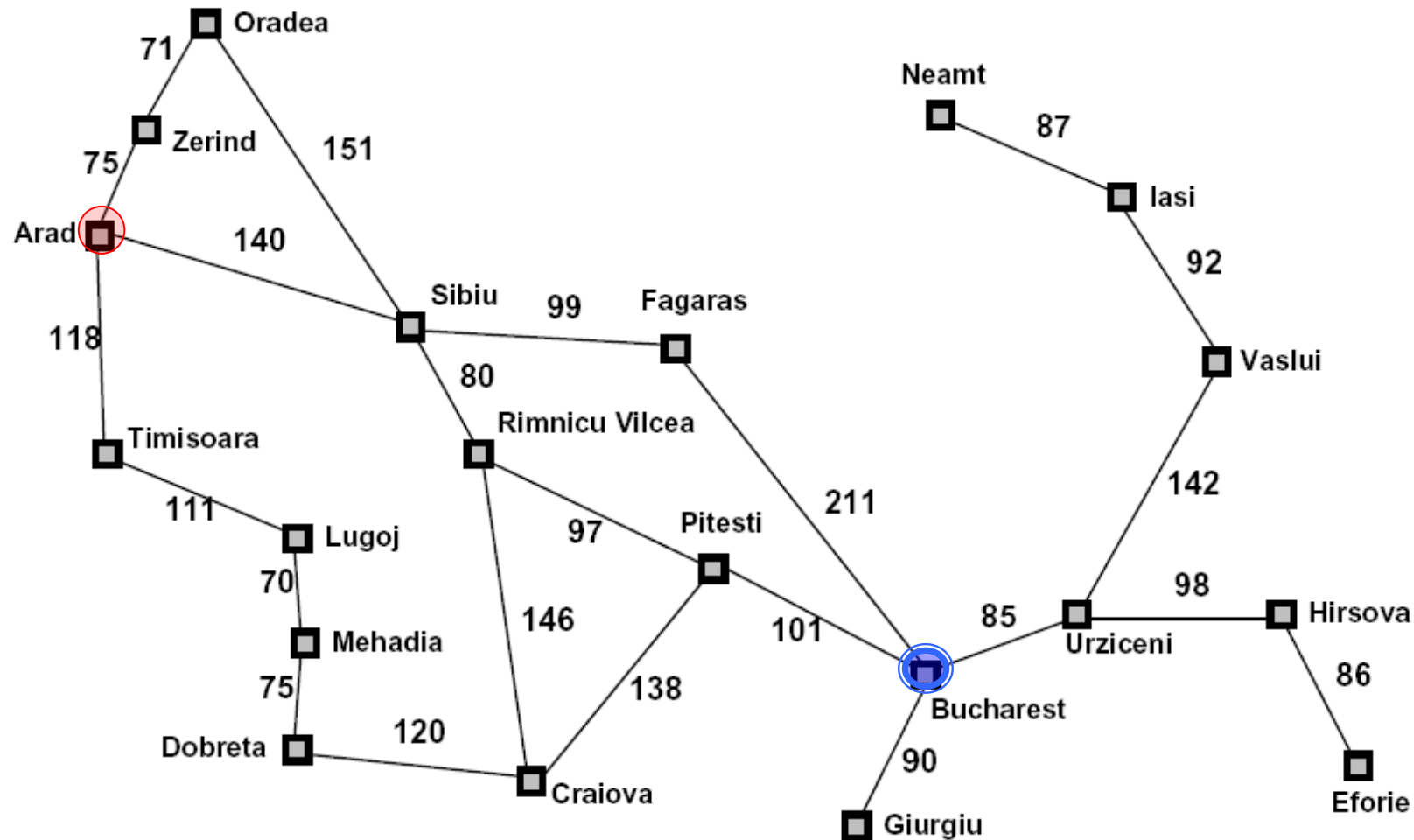
**Heuristics**

Sumber: Sergey Levine & Stuart Russell, University of California, Berkeley



IF

# Contoh: Fungsi Heuristik (Romania problem)



Straight-line distance  
to Bucharest

Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

$h(x)$

Sumber: Sergey Levine & Stuart Russell, University of California, Berkeley



# IF Contoh: Fungsi Heuristik (8-puzzles problem)

7	2	4
5		6
8	3	1

Start State



	1	2
3	4	5
6	7	8

Goal State

$h(x)$ : banyaknya angka yang salah penempatan berdasarkan *state* tujuan

# Tree-Search & Graph-Search Pseudocode

```
function TREE-SEARCH(problem, fringe) return a solution, or failure
  fringe ← INSERT(MAKE-NODE(INITIAL-STATE[problem]), fringe)
  loop do
    if fringe is empty then return failure
    node ← REMOVE-FRONT(fringe)
    if GOAL-TEST(problem, STATE[node]) then return node
    for child-node in EXPAND(STATE[node], problem) do
      fringe ← INSERT(child-node, fringe)
    end
  end
```

```
function GRAPH-SEARCH(problem, fringe) return a solution, or failure
  closed ← an empty set
  fringe ← INSERT(MAKE-NODE(INITIAL-STATE[problem]), fringe)
  loop do
    if fringe is empty then return failure
    node ← REMOVE-FRONT(fringe)
    if GOAL-TEST(problem, STATE[node]) then return node
    if STATE[node] is not in closed then
      add STATE[node] to closed
      for child-node in EXPAND(STATE[node], problem) do
        fringe ← INSERT(child-node, fringe)
      end
    end
  end
```



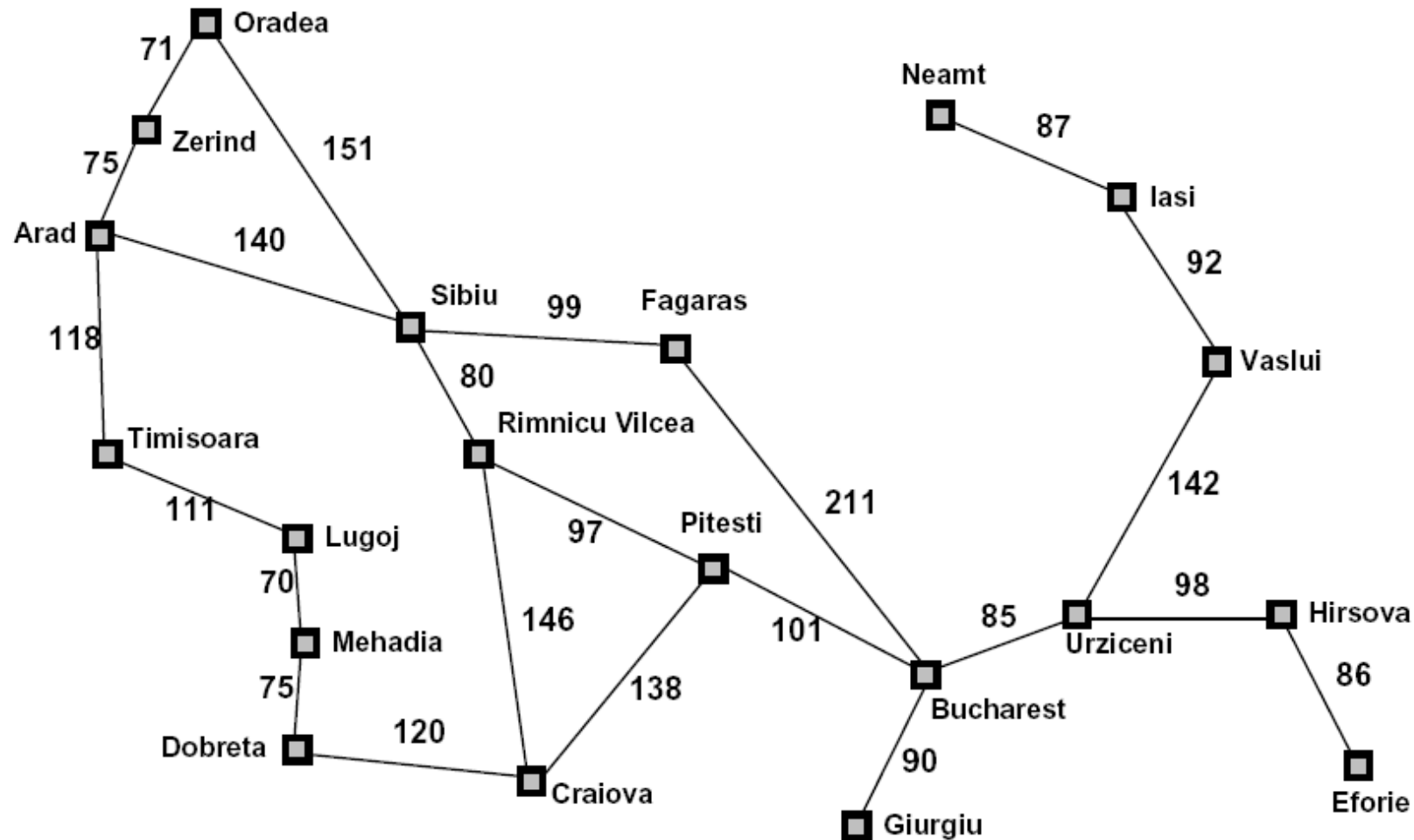


# Greedy Best First Search

- Best First Search mengekspand node yang mendekati goal
- Evaluation Function  $h(n)$  (Heuristics)
  - ➔ Estimasi *cost* dari *state*  $n$  ke *goal state*
- Misalnya,  $h_{SLD}(n)$  = Straight-Line Distance (jarak lurus)
- Kasus secara umum:
  - Best-first search langsung menuju goal (yang salah)



# Contoh: Romania problem

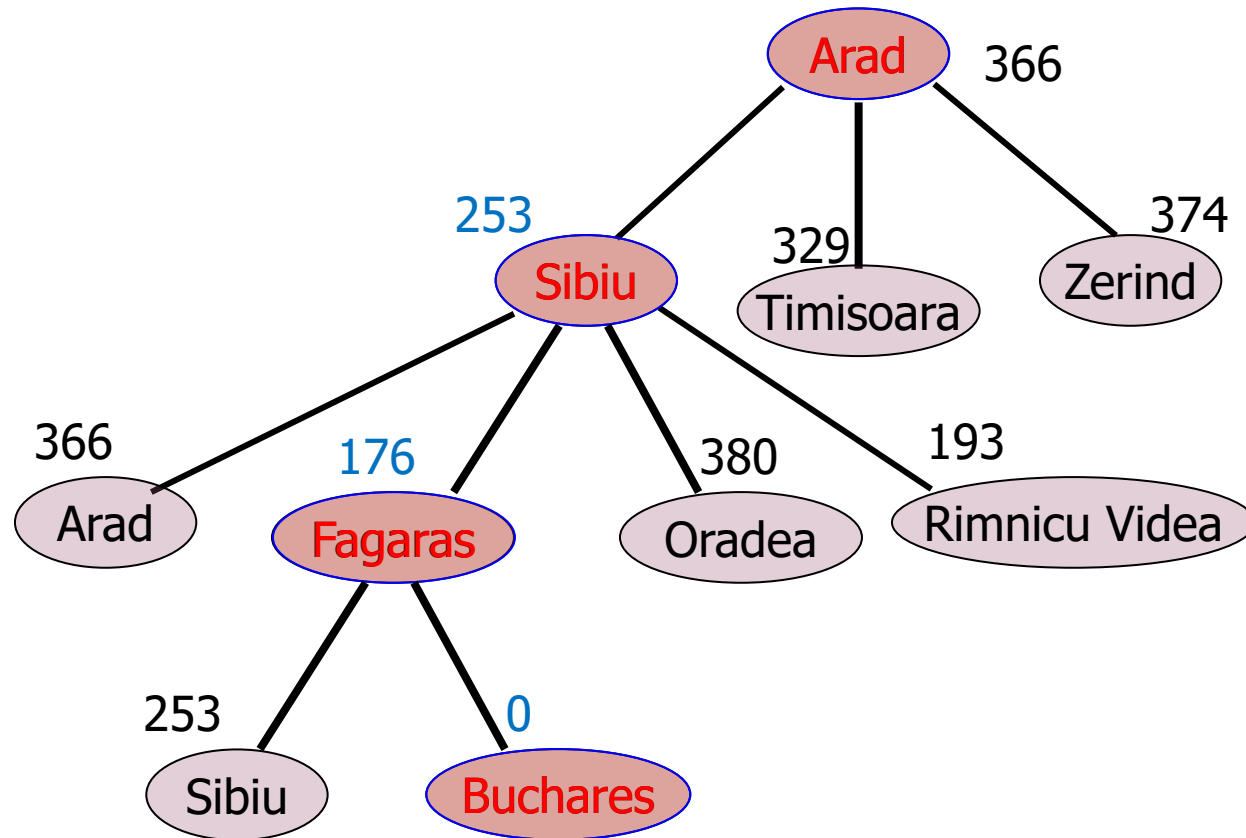


Straight-line distance  
to Bucharest

Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

$h(x)$

# Contoh: Greedy Best First Search (Romania problem)





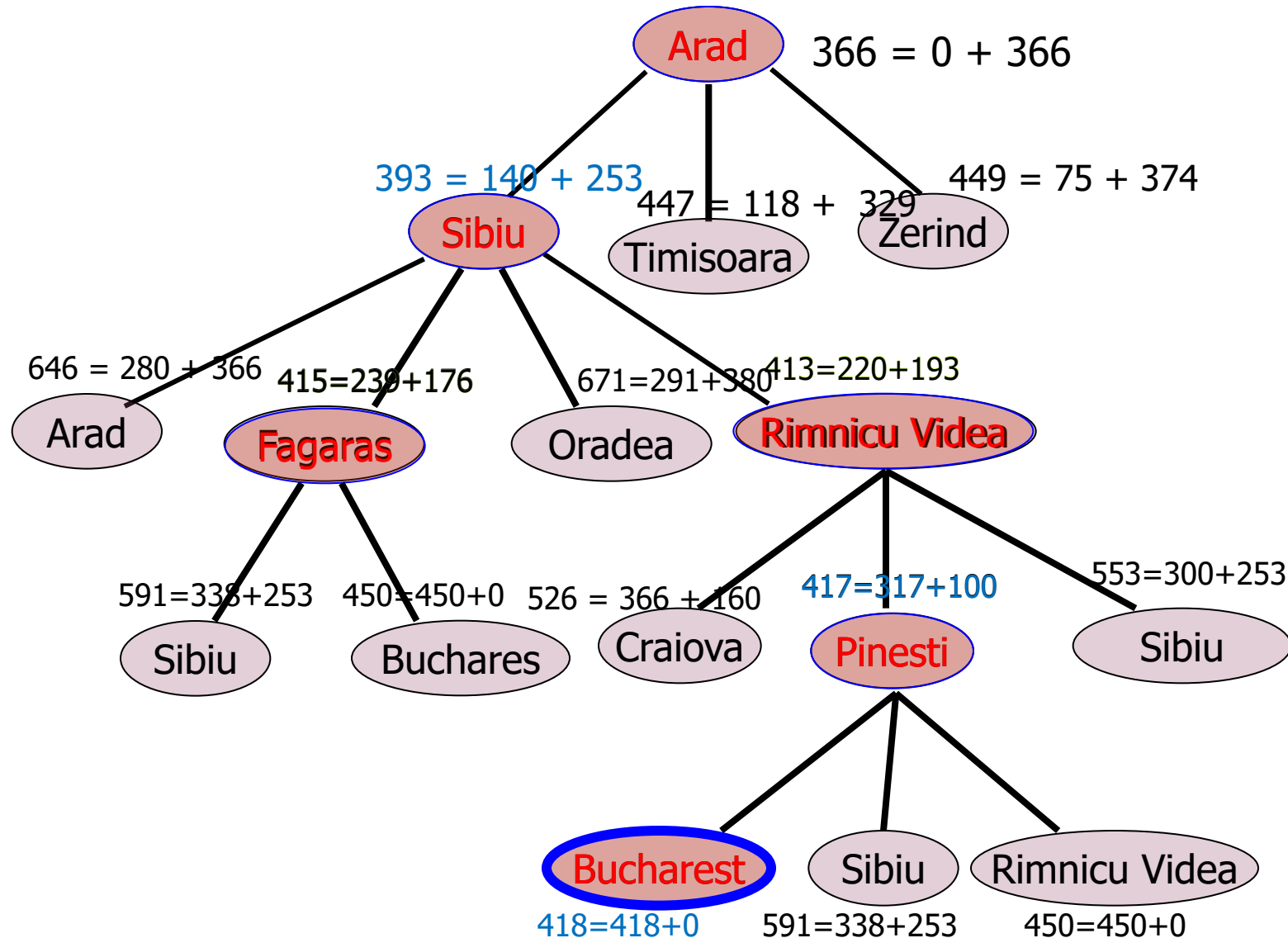
# Greedy Best First Search

- **Complete?**
  - Tidak, bisa terjadi looping, misal : Oradea sebagai goal : Iasi → Neamt → Iasi → Neamt ...
- **Time?**
  - $O(b^m)$  namun dengan heuristik yang baik akan memberikan perbaikan yang besar
- **Space?**
  - $O(b^m)$  Setiap node disimpan dalam memory
- **Optimal?**
  - Tidak, mestinya tidak melalui Fagaras untuk mencapai optimalnya

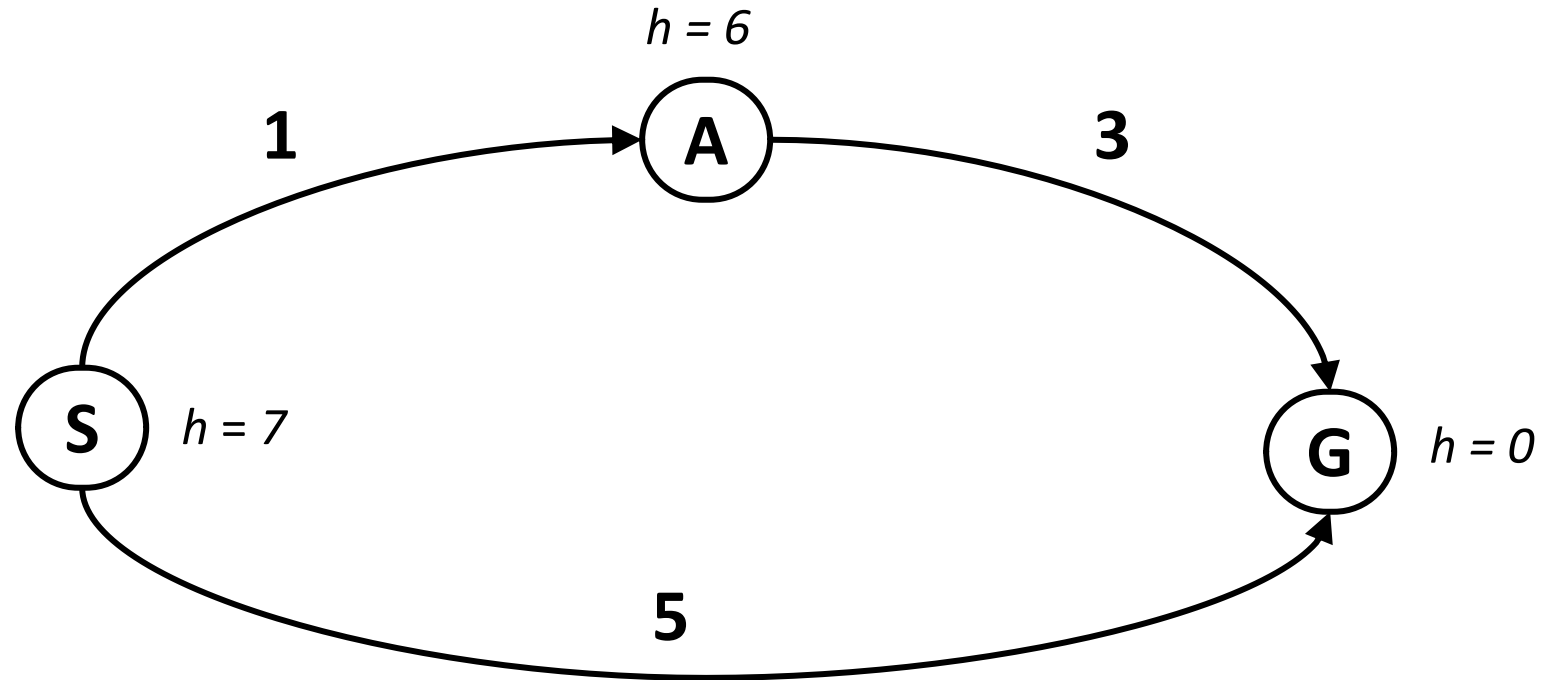
- Ide : menghindari untuk *expand path* yang memerlukan biaya besar
- Kombinasi UCS dan Greedy Search
- *Evaluation Function* :  $f(n) = g(n) + h(n)$ 
  - $g(n)$  = Cost yang dicapai sampai di state  $n$
  - $h(n)$  = Estimasi cost untuk sampai ke *goal* dari  $n$
  - $f(n)$  = Estimasi *total cost* dari *path*  $n$  sampai *goal*



# Contoh: A\* Search (Romania problem)



# Apakah A\* optimal?



- Tidak Berhasil jika *actual goal cost* < *estimated goal cost*
- Diperlukan estimasi *cost* yang kurang dari *actual cost*!

# Admissible Heuristic

- $A^*$  : *admissible heuristic*  $\rightarrow$  tidak *overestimate* jika

$$0 \leq h(n) \leq h^*(n)$$

- dimana  $h^*(n)$  adalah actual cost terkecil dari  $n$  ke goal
  - $h(n) \geq 0$  sehingga  $h(G) = 0$  untuk goal  $G$
  - Contoh,  $h_{SLD}(n)$  tidak overestimate terhadap jarak pada jalan sebenarnya
- 
- $A^*$  search  $\rightarrow$  Optimal



# Optimality of A\* Tree Search

Asumsi:

- A adalah optimal goal node
- B adalah suboptimal goal node
- $h$  is *admissible*

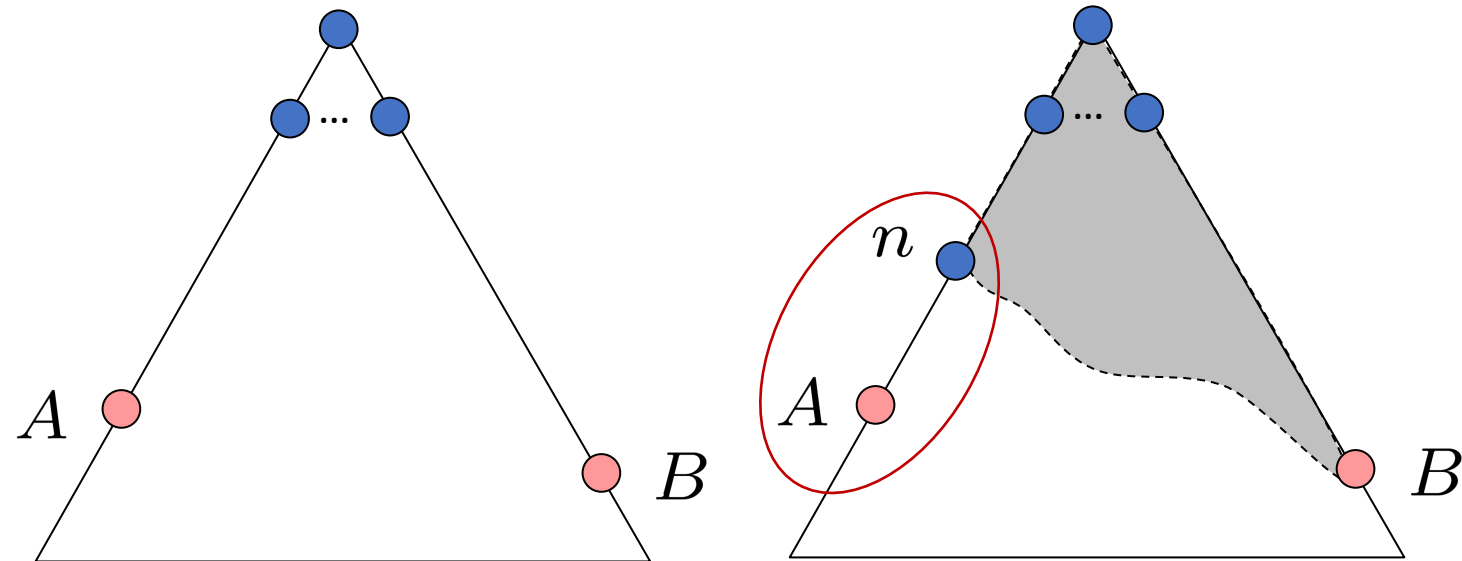
Claim:

A akan keluar dari fringe sebelum B

Proof:

- Jika B ada di fringe
- Beberapa *ancestor*  $n$  dari A ada di fringe juga
- Claim:  $n$  akan diexpand sebelum B

1.  $f(n)$  lebih kecil atau sama dengan  $f(A)$



$f(n) = g(n) + h(n)$  Definition of  $f$ -cost

$f(n) \leq g(A)$

Admissibility of  $h$

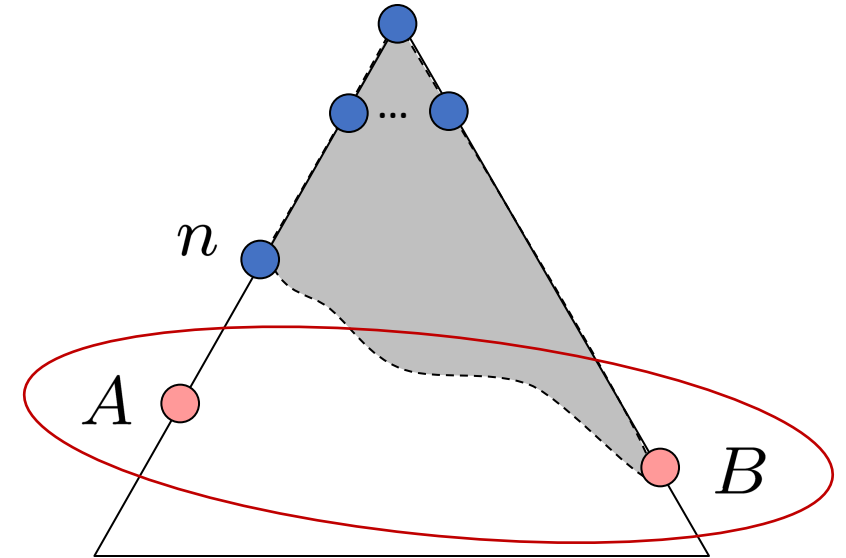
$g(A) = f(A)$

$h = 0$  at a goal

# Optimality of A\* Tree Search

Proof:

- Jika B ada di fringe
- Beberapa *ancestor*  $n$  dari A ada di fringe juga
- Claim:  $n$  akan diexpand sebelum B
  1.  $f(n)$  lebih kecil atau sama dengan  $f(A)$
  2.  $f(A)$  lebih kecil dari  $f(B)$



$$g(A) < g(B)$$

*B is suboptimal*

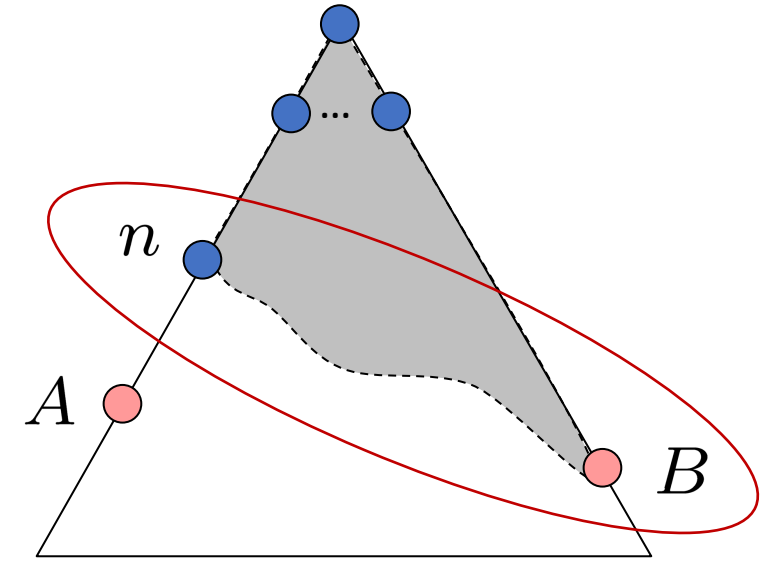
$$f(A) < f(B)$$

*$h = 0$  at a goal*

# Optimality of A\* Tree Search

Proof:

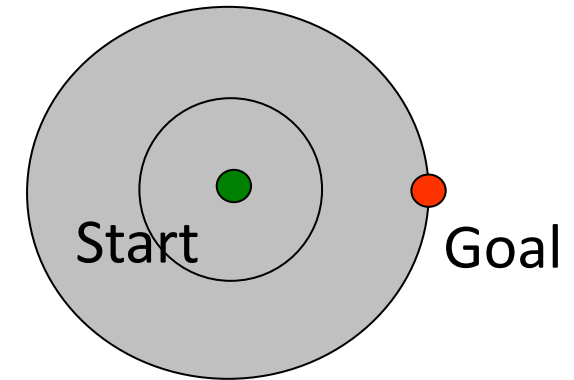
- Jika B ada di fringe
- Beberapa *ancestor*  $n$  dari A ada di fringe juga
- Claim:  $n$  akan diexpand sebelum B
  1.  $f(n)$  lebih kecil atau sama dengan  $f(A)$
  2.  $f(A)$  lebih kecil dari  $f(B)$
  3.  $n$  diexpand sebelum B
- Semua *ancestor* dari A diexpand sebelum B
- A diexpand sebelum B
- A\* search adalah optimal



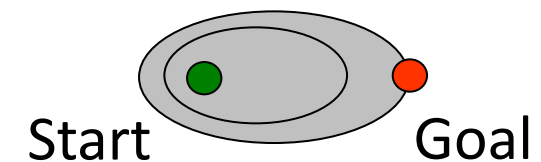
$$f(n) \leq f(A) < f(B)$$

# UCS vs A\* Contours

Uniform-cost search *expand* semua  
“*directions*”

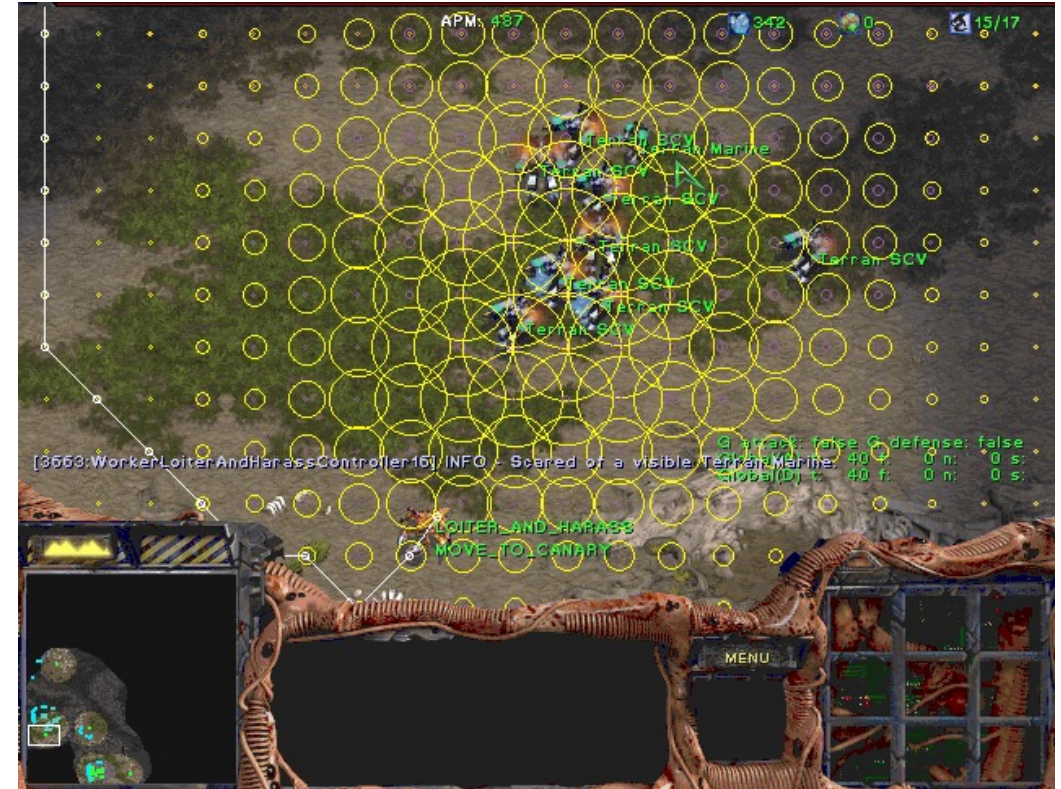


A\* search *expand* hanya arah menuju  
*goal*, tetapi tetap memastikan  
keoptimalan



- Complete ?
  - Ya, selama jumlah node  $f \leq f(G)$  terbatas
- Time ?
  - Exponensial
- Space ?
  - Setiap node disimpan dalam memory
- Optimal ?
  - Ya
    - A\* mengekspond node-node dengan  $f(n) < C^*$
    - A\* mengekspond beberapa node dengan  $f(n) = C^*$
    - A\* tidak akan mengekspond node dengan  $f(n) > C^*$

- Video games
- Pathing / routing problems
- Resource planning problems
- Robot motion planning
- Language analysis
- ...





# Konsistensi Fungsi Heuristik

- Fungsi Heuristic  $h(n)$  dikatakan konsisten jika setiap node  $n$  dan setiap successor  $n'$  dari  $n$  yang digenerate aksi  $a$ , maka estimasi cost dari  $n$  sampai ke goal tidak lebih besar dari cost sampai step  $n'$  ditambah estimasi cost  $n'$  ke goal

$$h(n) \leq c(n, a, n') + h(n')$$

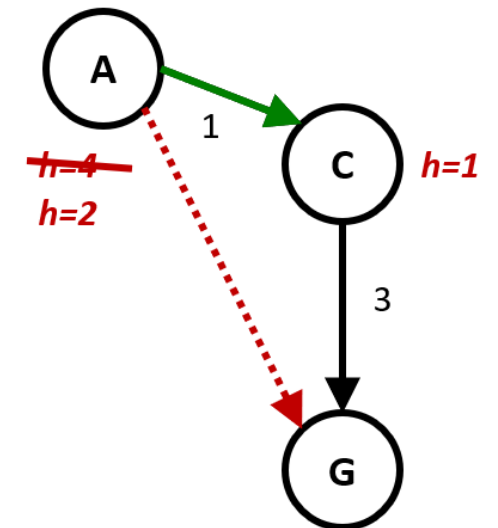
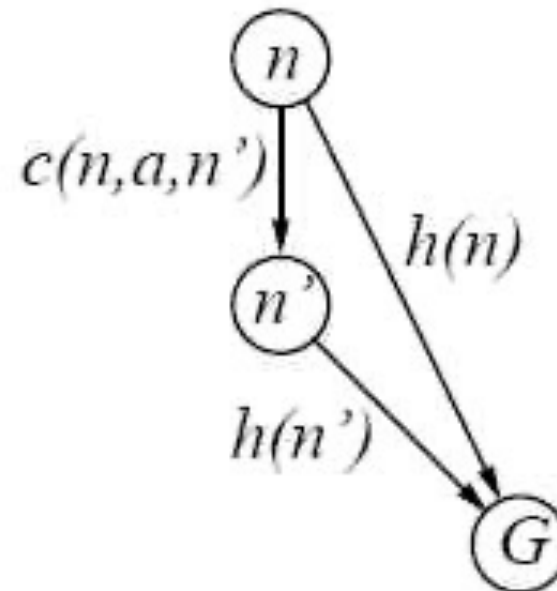
- Jika  $h(n)$  konsisten maka nilai dari  $f(n)$  melalui suatu path tidak berkurang

$$f(n') = g(n') + h(n')$$

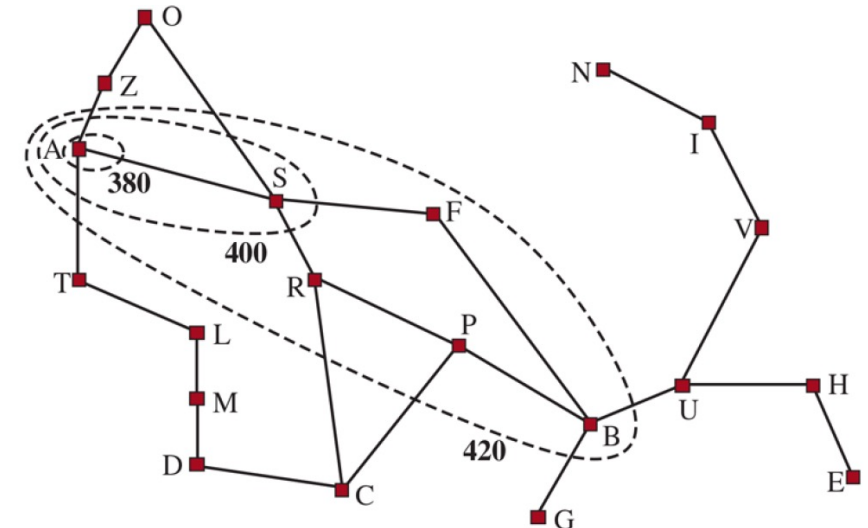
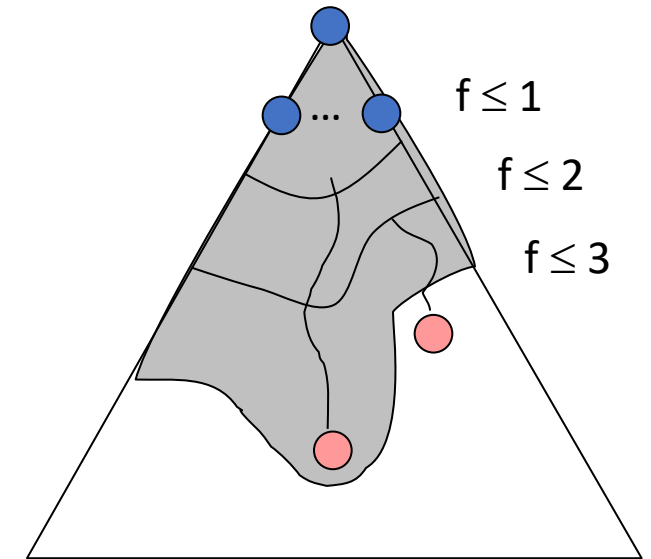
$$= g(n) + c(n, a, n') + h(n')$$

$$\geq g(n) + h(n)$$

$$= f(n)$$



- *Tree search:*
  - A\* adalah optimal jika *heuristic is admissible*
- *Graph search:*
  - A\* adalah optimal jika *heuristic is consistent*
- *Consistency implies admissibility*

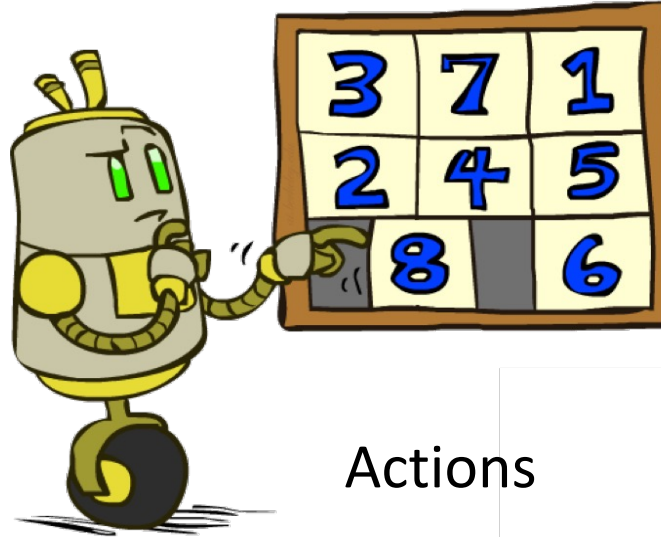




# Contoh: 8-puzzles

7	2	4
5		6
8	3	1

Start State



Actions

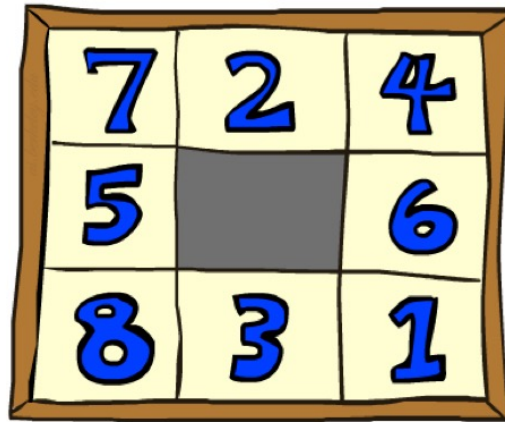
	1	2
3	4	5
6	7	8

Goal State

- Berapa banyak state? Rata-rata  $\rightarrow b = 3, d = 22$
- Apa saja aksinya?
- Berapa banyak *successors* dari *start state*? 4
- Bagaimana menentukan *cost*?

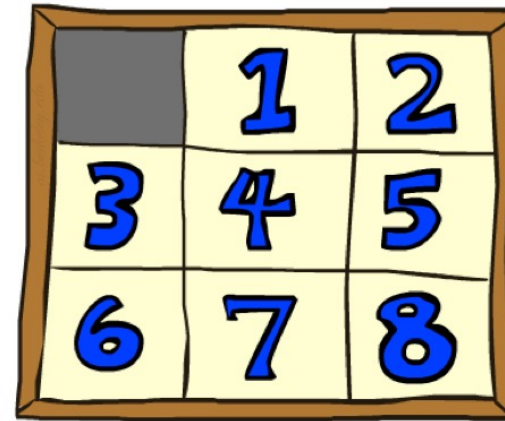
# Contoh: 8-puzzles I

- Heuristik: banyaknya kotak yang salah penempatan
- Mengapa *admissible*?
- $h(start) = 8$



7	2	4
5		6
8	3	1

Start State



	1	2
3	4	5
6	7	8

Goal State



# Contoh: 8-puzzles II

- Heuristik: *Manhattan distance* → jumlah jarak masing-masing kotak ke tujuan
- Mengapa *admissible*?
- $h(start) = 3 + 1 + 2 + 2 + 2 + 3 + 3 + 2 = 18$

7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

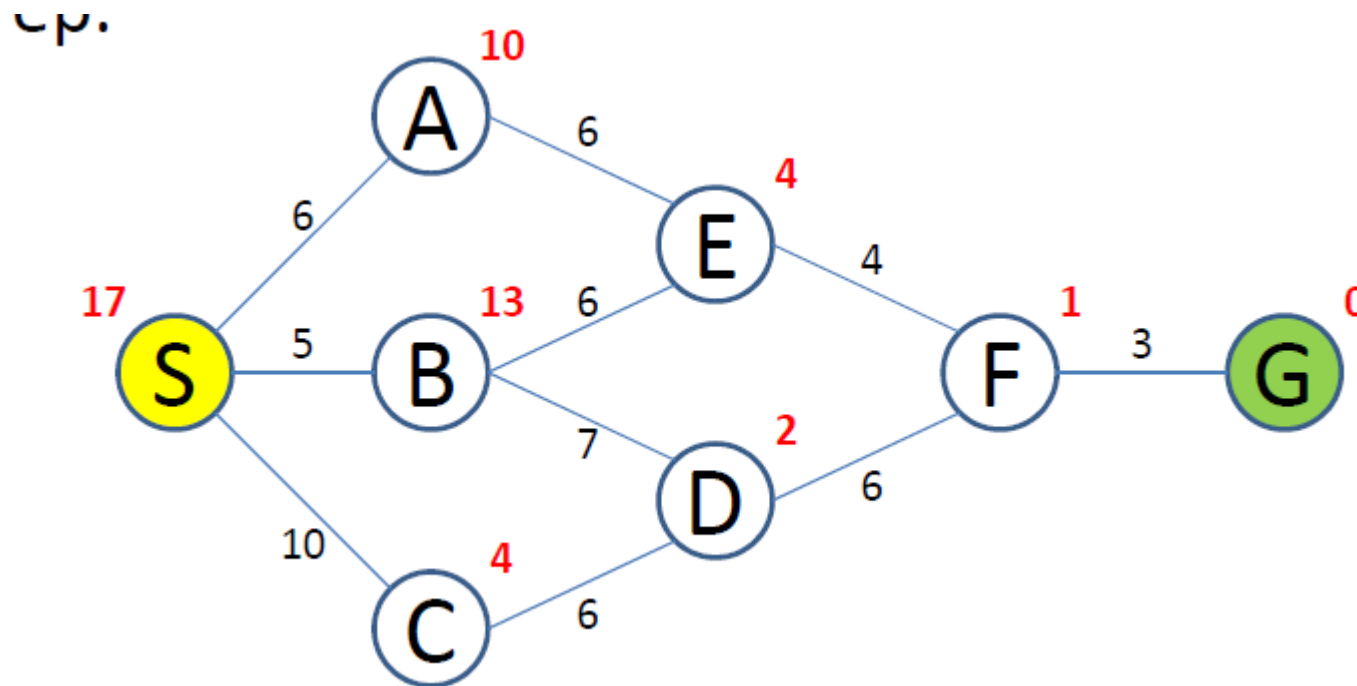
Goal State



- *Greedy Best-First*
- *A\**
- *Iterated Deepening A\* (IDA)*
- *Recursive Best-first Search (RBFS)*
- *Simplified Memory Bounded A\* (SMA)*

# Latihan

- Selesaikan menggunakan Greedy Best-First Search dan A\* Search



## Tugas 2 Individu

- Dikumpulkan: 8 Maret 2023
- Terdapat kasus 8-puzzles dengan informasi state awal dan goal sebagai berikut:

7	2	4
5		6
8	3	1

Start State

	1	2
3	4	5
6	7	8

Goal State

- Selesaikan kasus 8-puzzles diatas menggunakan 2 metode *informed search* dengan *actual cost* adalah satu setiap aksi dan fungsi heuristik adalah *Manhattan distance* (jumlah jarak masing-masing kotak ke tujuan) atau jumlah kotak salah tempat

## Tugas 3 Kelompok

- Deadline pengumpulan 22 Maret 2023
- Buat implementasi program dua algoritma *informed search* pada satu contoh kasus dan buat analisis perbandingan dari hasil solusi kedua algoritma tersebut.





**- TERIMA KASIH -**



**Teknik Informatika**  
department of informatics  
Fakultas Teknologi Informasi



[www.its.ac.id](http://www.its.ac.id)



[its\\_campus](#)



[institut teknologi sepuluh nopember](#)