# Informed search strategies

Informed search strategies (**Heuristic search** ) use problem–specific knowledge beyond the definition of the problem itself.

## History Of "Heuristic"

The word "heuristic" is derived from the Greek verb *heuriskein*, meaning "to find". Archimedes is said to have run naked down the street shouting *"Heureka"* (I have found it) after discovering the principle of flotation in his bath. Later generations converted this to Eureka.

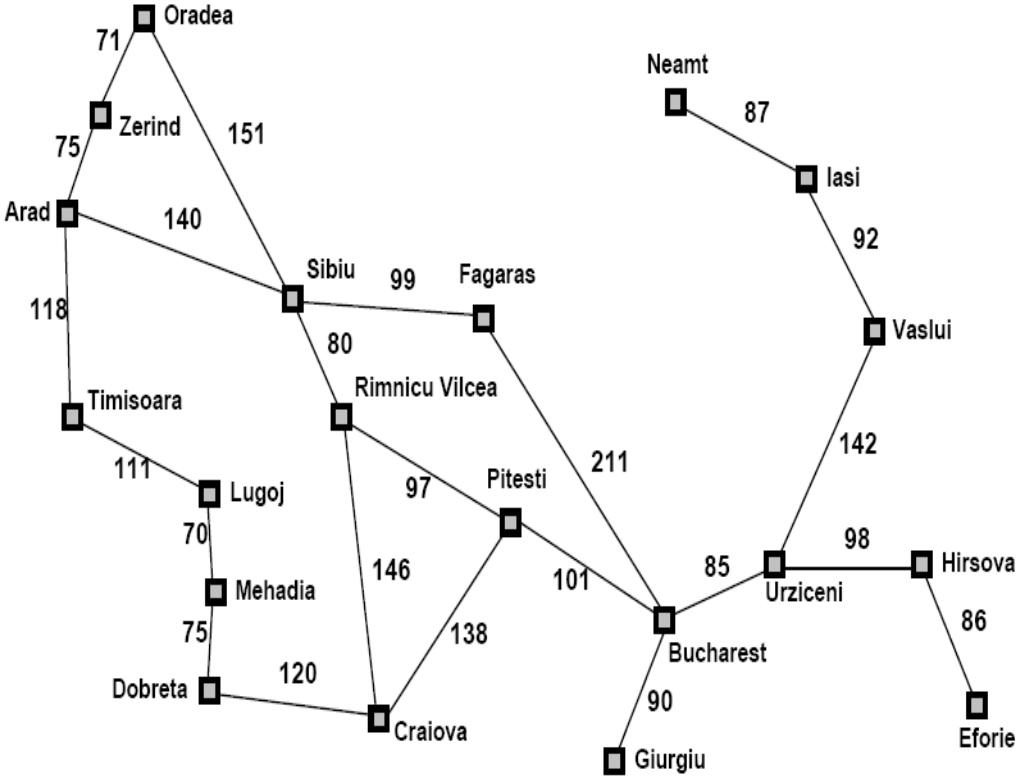
In 1957, George Polya a book called *How to Solve It* that used "heuristic" to refer to the study of methods for discovering and inventing problem solving techniques Two definitions of Artificial Intelligence are:

| **Algorithms** | **Heuristics** |
|---|---|
| Algorithms exhaust all possibilities before arriving at a solution.<br><br>Inefficient, unsophisticated and time consuming, but guarantee solutions.<br><br>If we were to unscramble these letters to form a word, using an algorithm would take 907,208 possibilities.<br>S P L O Y O C H Y G | Heuristics make it easy for us to use simple principles to arrive at solutions to problems.<br><br>Do not guarantee solution.<br><br>Try putting consonants at the beginning and Y at the end.<br>S P L O Y O C H Y G<br>P S Y C H O L O G Y |

Best-first search cab be recognized in two types :

- greedy search
- A* search

**heuristic function h(n):**
estimated cost of the cheapest path from node n to a goal node. If *n* is a goal node, then h(*n*)=0.

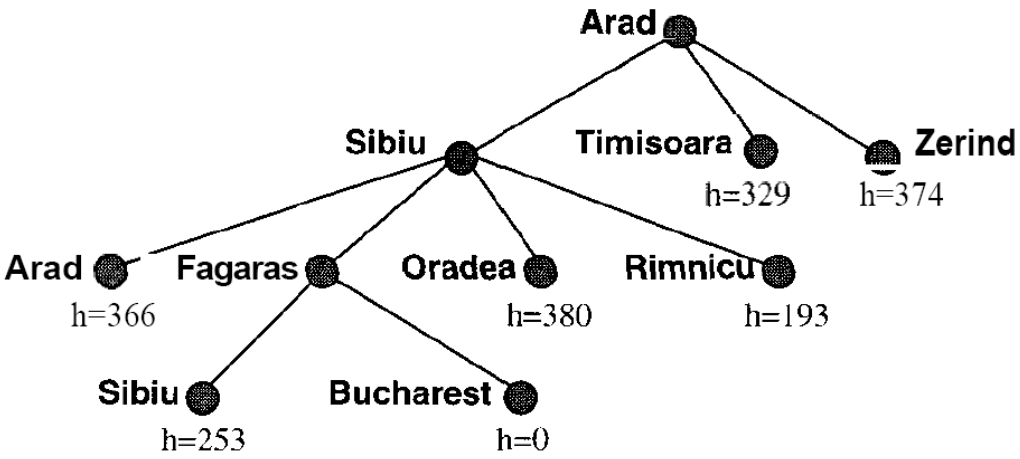



Here are straight-line distances between each node and the goal location. **If not given for a problem, you must calculate them using a ruler, or define your own heuristic.**

| 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 |

Note that in greedy search we will have :

$$f(n) = h(n)$$



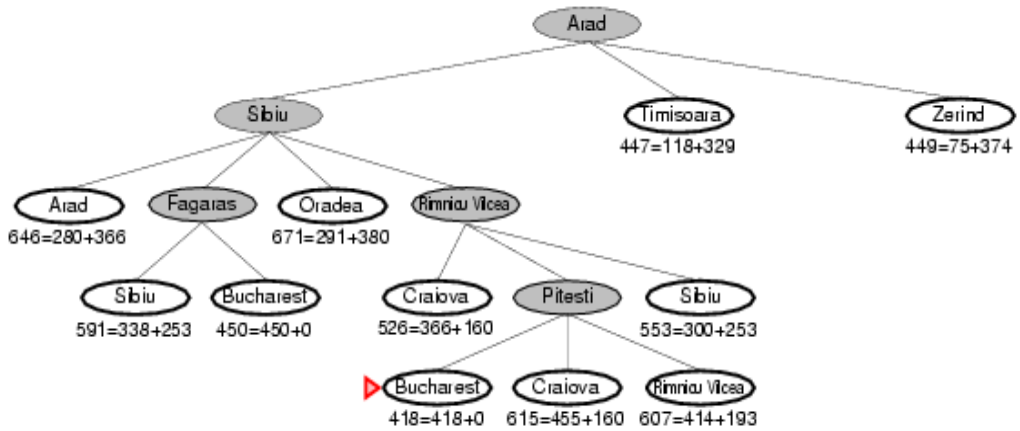

## A* search

The Idea is to avoid expanding paths that are already expensive.
Now, Evaluation function equals

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

where

**g(n)** = cost so far to reach *n*
**h(n)** = estimated the cheapest cost from *n* to goal
**f(n)** = estimated total cost of path through *n* to goal




**Optimality of A***
**Theorem**: If *h(n)* is admissible, A* TREE-SEARCH is optimal.
**Theorem:** If *h(n)* is consistent, A* GRAPH- SEARCH is optimal.

**Admissible heuristics**
A heuristic *h(n)* is **admissible** if for every node *n*, $h(n) \leq h^*(n)$, where $h^*(n)$ is the true cost to reach the goal state from *n*.

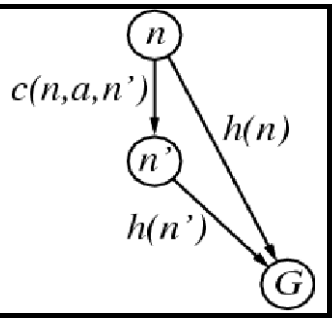
An admissible heuristic **never overestimates** the cost to reach the goal, i.e., it is optimistic. Example: $h_{SLD}(n)$

**Consistent (Monotonic) Heuristics**

A heuristic h(n) is consistent or monotonic if for every node *n*, every successor *n'* of *n* generated by any action *a*

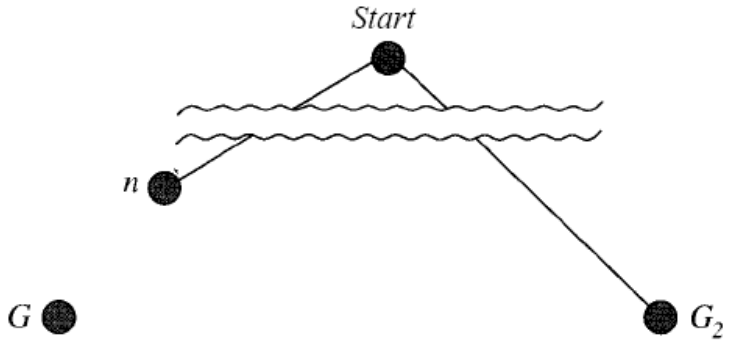
***h(n) ≤ c(n,a,n') + h(n')***

**Triangle inequality**




**Optimality of A* (TREE-search) Informal Proof**

Suppose some **suboptimal goal *G2*** has been generated and is in the fringe. Let *n* be an unexpanded node in the fringe such that *n* is on a shortest path to an **optimal goal *G***




f(G2) > f(G) from above
h(n) ≤ h*(n) since h is admissible
g(n) + h(n) ≤ g(n) + h*(n)
f(n) ≤ f(G)

Hence *f(G2) > f(n)*, and A* will never select G2 for expansion

**First : Optimality of A* (TREE-search)**

Suppose some suboptimal goal $G_2$ has been generated and is in the fringe. Let *n* be an unexpanded node in the fringe such that *n* is on a shortest path to an optimal goal *G*

(i)

$f(G_2) = g(G_2)+h(G_2)$
$= g(G_2)$ since $h(G_2) = 0$
$> C^*$ since $G_2$ is suboptimal then $g(G_2) > g(G)$

(ii)

If h(n) is admissible so $h(n) \le h^*(n)$
$f(n) = g(n)+h(n)$
$\le g(n) + h^*(n)$
$\le C^*$

from(i) and (ii)

$f(n) \le C^* < f(G2)$

So, $G_2$ will not be expanded and A* must return n

**Second : Optimality of A* (GRAPH-search)**

If *h* is consistent, we have $f(n') = g(n') + h(n')$

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

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

i.e., *f(n)* is **non-decreasing** along any path.

## Memory-bounded heuristic search (A* Variations)

Some solutions to A* **space problems**, maintaining completeness and optimality.

**1- Iterative-deepening A* ( IDA*)**

Here cutoff information is the *f*-cost ($g + h$) instead of depth

**2- Recursive best-first search(RBFS)**

Recursive algorithm that attempts to mimic standard best-first search with linear space.

- Space complexity is *O(bd)*.

**3- (simple) Memory-bounded A* ((S)MA*)**

Drop the worst-leaf node when memory is full

## Heuristic quality

**1- Effective branching factor b***

Effective branching factor is the branching factor that a uniform tree of depth *d* would have in order to contain *N+1* nodes.

$$N + 1 = 1 + b^* + (b^*)^2 + ... + (b^*)^d$$

A good value of b* is 1.

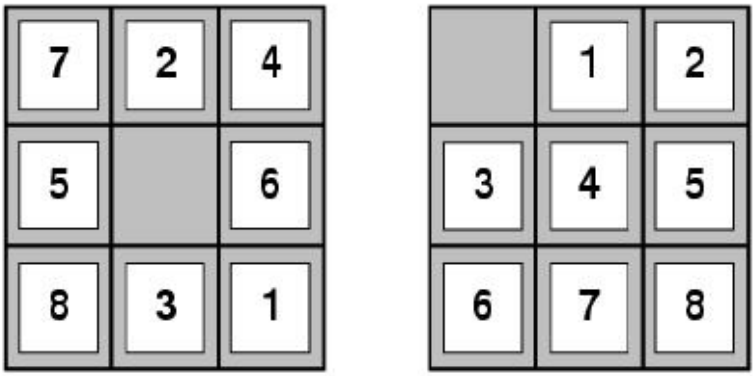
**2- dominance**

If $\mathbf{h_2(n) >= h_1(n)}$ for all *n* (both admissible) then $\mathbf{h_2}$ **dominates** $\mathbf{h_1}$ and is better for search

For the 8-puzzle knows two commonly used heuristics:

- the number of misplaced tiles
- sum of the distances of the tiles from their goal positions (manhattan distance)

Example:




Start State Goal State

$h_1$ = the number of misplaced tiles

$h_1(s)=8$

$h_2$ = manhattan distance.

$h_2(s)=3+1+2+2+2+3+3+2=18$

Comments

$h_2(n) >= h_1(n)$ for all *n* (both admissible)

$h_2$ dominates $h_1$

$h_2$ is better for search than $h_1$

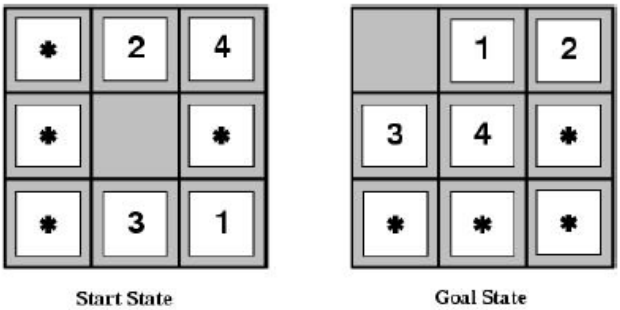
## Inventing admissible heuristics

1- Admissible heuristics can be derived from the exact solution cost of a **relaxed version of the problem** (A problem with fewer restrictions on the actions) :

- Relaxed 8-puzzle for *h1* : **a tile can move anywhere**
- Relaxed 8-puzzle for *h2* : **a tile can move to any adjacent square**

**ABSolver** found a useful heuristic for the **rubic cube**.

2- Admissible heuristics can also be derived from the **solution cost of a sub-problem** of a given problem.
This cost is a lower bound on the cost of the real problem.




Start State Goal State

3- Another way to find an admissible heuristic is through **learning from experience**:
Experience = solving lots of 8-puzzles

## Heuristics in mathematics

3 | n

If the sum of all the digits of n down to a single digit equals 3, 6, or 9, then 3 divides n.

For example, is 17,587,623 divisible by 3?

1 + 7 + 5 + 8 + 7 + 6 + 2 + 3 = 39

3 + 9 = 12

1 + 2 = 3 → YES! 3 divides 17,587,623

7 | n

1. Remove least significant digit from n and multiply it by two.
2. Subtract the doubled number from the remaining digits.
3. If result is divisible by 7, the original number is divisible by 7
4. Repeat if unable to determine from result.

Examples of checking for divisibility by 7

1,876 → 187 – 12 = 175 → 17 – 10 = 7 ✓

4,923 → 492 – 6 = 486 → 48 – 12 = 36 ×

34,461→3,446 –2= 3,444 → 344 –8 = 336 → 33 –12 = 21✓

11 | n

1. Starting with the most significant digit of n, adding the first digit, subtracting the next digit, adding the third digit, subtracting the fourth, and so on.
2. If the result is 0 or a multiple of 11, then the original number is divisible by 11.
3. Repeat if unable to determine from result.

Examples of checking for divisibility by 11

285311670611 → 2 –8 +5 –3 +1 –1 +6 –7 + 0– 6 + 1 – 1 = –11 ✓

279048 → 2 – 7 + 9 – 0 + 4 – 8 = 0 ✓

## Heuristics in business

- heuristics tend to be **domain-specific** .
- we find heuristics as a tool for **usability in games**.
- with **mobile devices and services**,
- **Discovering Musical Patterns** through Heuristics.
- **Economics** has been relying on quantitative models
- **Programming** also has heuristics.
- **Modern anti-viruses** such as ESET NOD32 uses heuristics to detect unknown viruses.
- A **heuristic evaluation** is a usability method for **HP PhotoSmart printer**. This Heuristic Evaluation covers the touch screen display, selecting pictures for Printing.