Basics of AI and Machine Learning

State-Space Search: Best-first Graph Search

Jendrik Seipp

Linköping University

Slides modified from Basel AI group, with permission

Best-first Search

Best-first Search

Best-first search is a class of search algorithms that expand the "most promising" node in each iteration.

- decision which node is most promising uses heuristics. . .
- . . . but not necessarily exclusively.

- implementation essentially like uniform cost search
- different choices of f $\rightsquigarrow$ different search algorithms

Best-first Search

Best-first search is a class of search algorithms that expand the "most promising" node in each iteration.

- decision which node is most promising uses heuristics. . .
- . . . but not necessarily exclusively.

Best-first Search

A best-first search is a heuristic search algorithm that evaluates search nodes with an evaluation function f and always expands a node n with minimal $f(n)$ value.

- implementation essentially like uniform cost search
- different choices of f $\rightsquigarrow$ different search algorithms

The Most Important Best-first Search Algorithms

the most important best-first search algorithms:

The Most Important Best-first Search Algorithms

the most important best-first search algorithms:

- $f(n) = h(n.\text{state})$: greedy best-first search
  $\rightsquigarrow$ only the heuristic counts

The Most Important Best-first Search Algorithms

the most important best-first search algorithms:

- $f(n) = h(n.\text{state})$: greedy best-first search
  $\rightsquigarrow$ only the heuristic counts
- $f(n) = g(n) + h(n.\text{state})$: A*
  $\rightsquigarrow$ combination of path cost and heuristic

The Most Important Best-first Search Algorithms

the most important best-first search algorithms:

- $f(n) = h(n.\text{state})$: greedy best-first search
  $\rightsquigarrow$ only the heuristic counts
- $f(n) = g(n) + h(n.\text{state})$: A*
  $\rightsquigarrow$ combination of path cost and heuristic
- $f(n) = g(n) + w \cdot h(n.\text{state})$: weighted A*
  $w \in \mathbb{R}_0^+$ is a parameter
  $\rightsquigarrow$ interpolates between greedy best-first search and A*

The Most Important Best-first Search Algorithms

the most important best-first search algorithms:

- $f(n) = h(n.\text{state})$: greedy best-first search
  $\rightsquigarrow$ only the heuristic counts
- $f(n) = g(n) + h(n.\text{state})$: A*
  $\rightsquigarrow$ combination of path cost and heuristic
- $f(n) = g(n) + w \cdot h(n.\text{state})$: weighted A*
  $w \in \mathbb{R}_0^+$ is a parameter
  $\rightsquigarrow$ interpolates between greedy best-first search and A*

What do we obtain with $f(n) := g(n)$? $\rightsquigarrow$ uniform cost search

Best-first Search: Graph Search or Tree Search?

Best-first search can be graph search or tree search.

- here: graph search (i.e., with duplicate elimination), which is the more common case

Best-first Search

Best-first Search

```
$open$ := new MinHeap ordered by $\langle f, h \rangle$
if $h(\text{init}()) < \infty$:
    $open$.insert(make_root_node())
$closed$ := new HashSet
while not $open$.is_empty():
    $n$ := $open$.pop_min()
    if $n$.state $\notin$ $closed$:
        $closed$.insert($n$.state)
        if is_goal($n$.state):
            return extract_path($n$)
        for each $\langle a, s' \rangle \in$ succ($n$.state):
            if $h(s') < \infty$:
                $n'$ := make_node($n, a, s'$)
                $open$.insert($n'$)
return unsolvable
```

Best-first Search: Properties

properties:

- complete if h is safe: duplicate detection
- optimality depends on f

Greedy Best-first Search

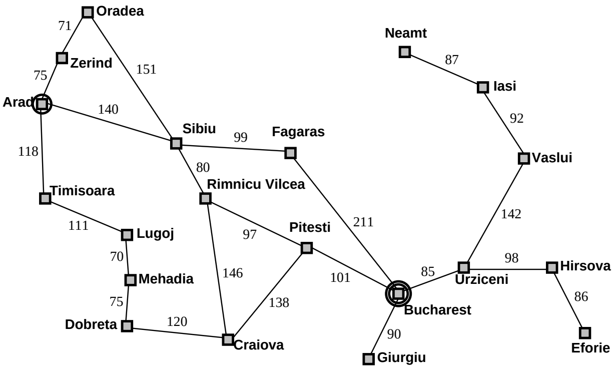
Greedy Best-first Search

Greedy Best-first Search

only consider the heuristic: $f(n) = h(n.\text{state})$

Note: usually without reopening (for reasons of efficiency)

Example: Greedy Best-first Search for Route Planning

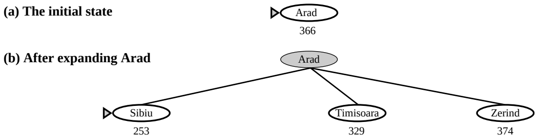


Arad	366
Bucharest	0
Craiova	160
Drobeta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

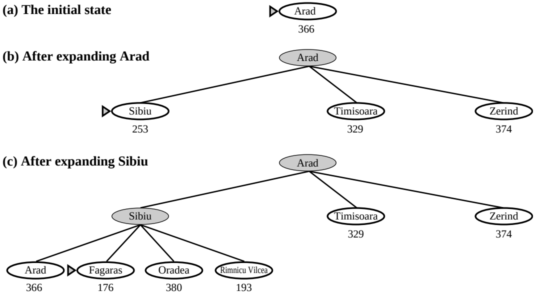
Example: Greedy Best-first Search for Route Planning

(a) The initial state

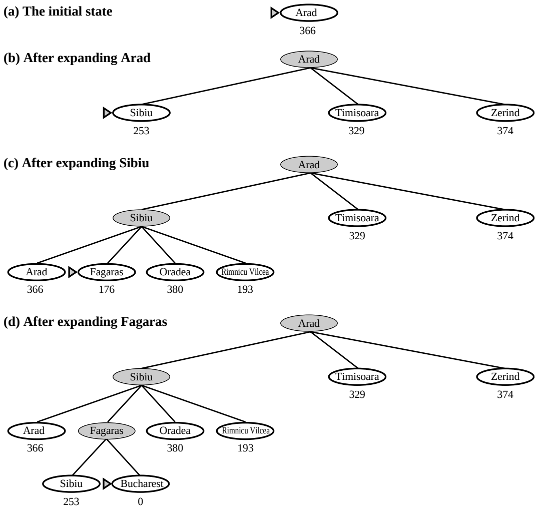
Example: Greedy Best-first Search for Route Planning



Example: Greedy Best-first Search for Route Planning



Example: Greedy Best-first Search for Route Planning



Greedy Best-first Search: Properties

- complete with safe heuristics
  (like all variants of best-first graph search)
- suboptimal: solutions can be arbitrarily bad
- often very fast: one of the fastest search algorithms in practice

A^*

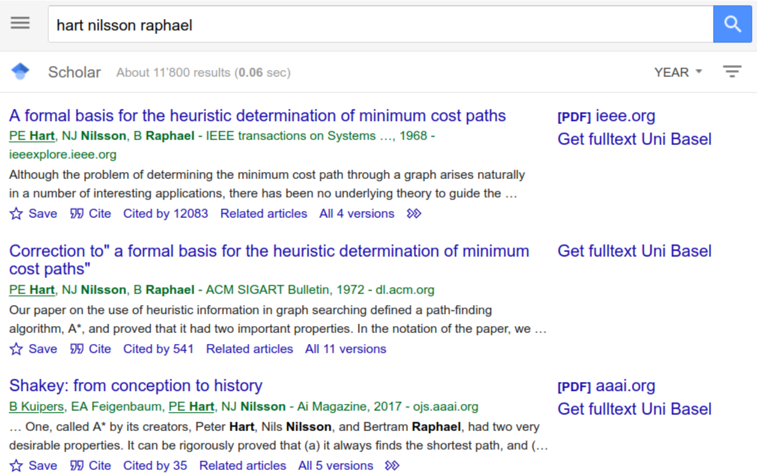
A*

A*

combine greedy best-first search with uniform cost search:
$f(n) = g(n) + h(n.\text{state})$

- trade-off between path cost and proximity to goal
- $f(n)$ estimates overall cost of cheapest solution from initial state via n to the goal

A*: Citations



hart nilsson raphael

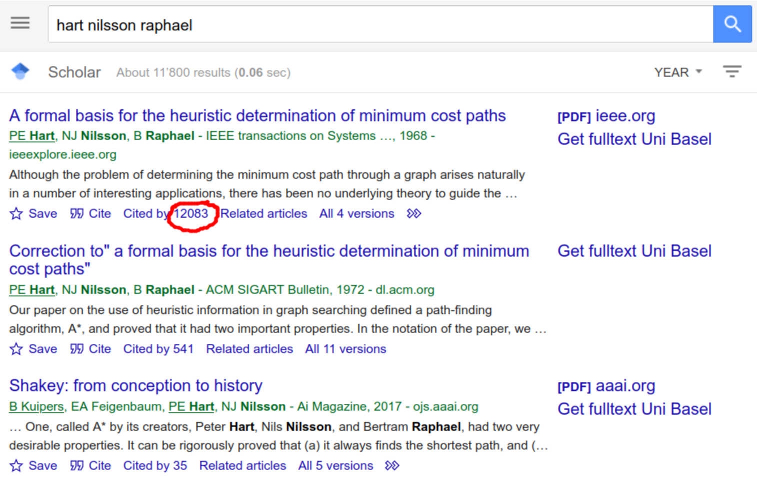
Scholar About 11'800 results (**0.06** sec) YEAR

A formal basis for the heuristic determination of minimum cost paths
PE **Hart**, NJ **Nilsson**, B **Raphael** - IEEE transactions on Systems …, 1968 - ieeexplore.ieee.org
Although the problem of determining the minimum cost path through a graph arises naturally in a number of interesting applications, there has been no underlying theory to guide the …
☆ Save 99 Cite Cited by 12083 Related articles All 4 versions
[PDF] ieee.org
Get fulltext Uni Basel

Correction to" a formal basis for the heuristic determination of minimum cost paths"
PE **Hart**, NJ **Nilsson**, B **Raphael** - ACM SIGART Bulletin, 1972 - dl.acm.org
Our paper on the use of heuristic information in graph searching defined a path-finding algorithm, A*, and proved that it had two important properties. In the notation of the paper, we …
☆ Save 99 Cite Cited by 541 Related articles All 11 versions
Get fulltext Uni Basel

Shakey: from conception to history
B Kuipers, EA Feigenbaum, PE **Hart**, NJ **Nilsson** - Ai Magazine, 2017 - ojs.aaai.org
… One, called A* by its creators, Peter **Hart**, Nils **Nilsson**, and Bertram **Raphael**, had two very desirable properties. It can be rigorously proved that (a) it always finds the shortest path, and (…
☆ Save 99 Cite Cited by 35 Related articles All 5 versions
[PDF] aaai.org
Get fulltext Uni Basel

A*: Citations



hart nilsson raphael

Scholar About 11'800 results (**0.06** sec) YEAR

A formal basis for the heuristic determination of minimum cost paths

PE **Hart**, NJ **Nilsson**, B **Raphael** - IEEE transactions on Systems ..., 1968 - ieeexplore.ieee.org

Although the problem of determining the minimum cost path through a graph arises naturally in a number of interesting applications, there has been no underlying theory to guide the ...

☆ Save Cite Cited by 12083 Related articles All 4 versions

[PDF] ieee.org
Get fulltext Uni Basel

Correction to" a formal basis for the heuristic determination of minimum cost paths"

PE **Hart**, NJ **Nilsson**, B **Raphael** - ACM SIGART Bulletin, 1972 - dl.acm.org

Our paper on the use of heuristic information in graph searching defined a path-finding algorithm, A*, and proved that it had two important properties. In the notation of the paper, we ...

☆ Save Cite Cited by 541 Related articles All 11 versions

Get fulltext Uni Basel

Shakey: from conception to history

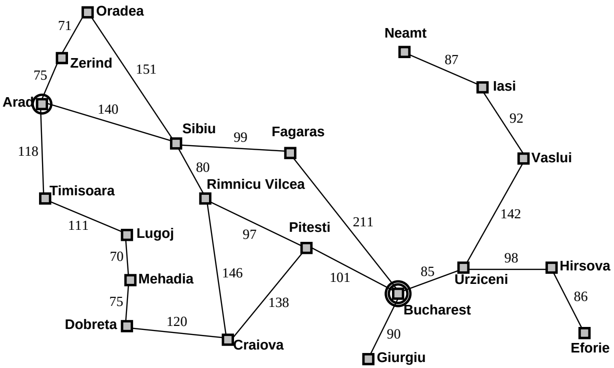
B Kuipers, EA Feigenbaum, PE **Hart**, NJ **Nilsson** - Ai Magazine, 2017 - ojs.aaai.org

... One, called A* by its creators, Peter **Hart**, Nils **Nilsson**, and Bertram **Raphael**, had two very desirable properties. It can be rigorously proved that (a) it always finds the shortest path, and (...

☆ Save Cite Cited by 35 Related articles All 5 versions

[PDF] aaai.org
Get fulltext Uni Basel

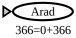
Example: A* for Route Planning



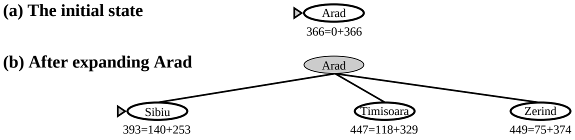
Arad	366
Bucharest	0
Craiova	160
Drobeta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Example: A* for Route Planning

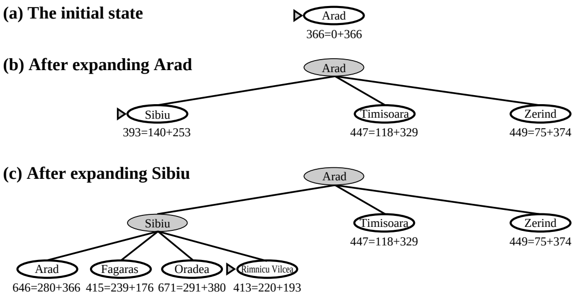
(a) The initial state



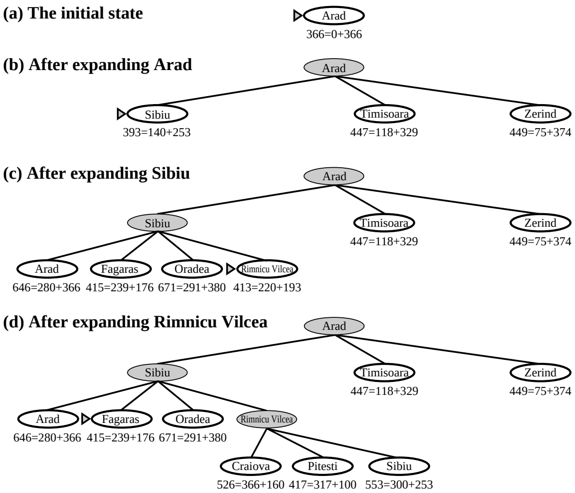
Example: A* for Route Planning



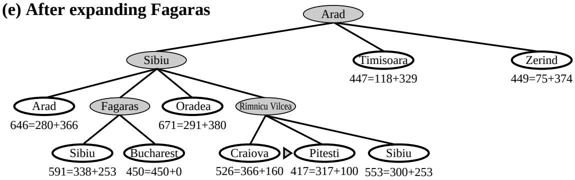
Example: A* for Route Planning



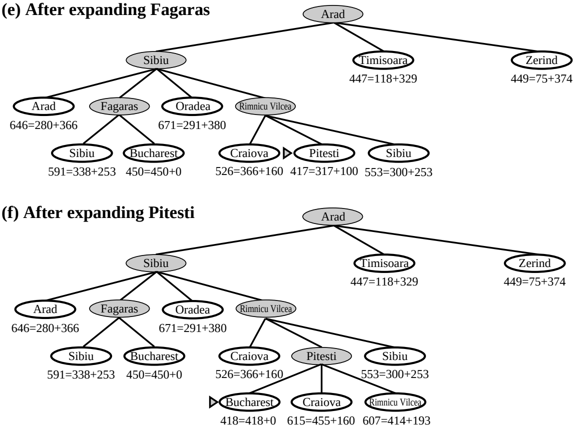
Example: A* for Route Planning



Example: A* for Route Planning



Example: A* for Route Planning



A*: Properties

- complete with safe heuristics
  (like all variants of best-first graph search)
- with reopening: optimal with admissible heuristics
- without reopening: optimal with heuristics
  that are admissible and consistent

Weighted A*

Weighted A*

Weighted A*

A* with more heavily weighted heuristic:

$f(n) = g(n) + w \cdot h(n.\text{state})$,

where weight $w \in \mathbb{R}_0^+$ with $w \geq 1$ is a freely choosable parameter

Weighted A^*: Properties

weight parameter controls "greediness" of search:

- $w = 0$: like uniform cost search
- $w = 1$: like A^*
- $w \to \infty$: like greedy best-first search

with $w \geq 1$ properties analogous to A^*:

- *h* admissible:
  found solution guaranteed to be at most w times as expensive as optimum when reopening is used

Summary

Summary

best-first graph search with evaluation function f:

- $f = h$: greedy best-first search
  suboptimal, often very fast
- $f = g + h$: A^*
  optimal if h admissible and consistent
- $f = g + w \cdot h$: weighted A^*
  for $w \geq 1$ suboptimality factor at most w
  under same conditions as for optimality of A^*