

# Foundations of Artificial Intelligence

## 19. State-Space Search: Properties of A\*, Part II

Malte Helmert

University of Basel

March 31, 2021

# Foundations of Artificial Intelligence

March 31, 2021 — 19. State-Space Search: Properties of A\*, Part II

19.1 Introduction

19.2 Monotonicity Lemma

19.3 Optimality of A\* without Reopening

19.4 Time Complexity of A\*

19.5 Summary

## State-Space Search: Overview

Chapter overview: state-space search

- ▶ 5.–7. Foundations
- ▶ 8.–12. Basic Algorithms
- ▶ 13.–19. Heuristic Algorithms
  - ▶ 13. Heuristics
  - ▶ 14. Analysis of Heuristics
  - ▶ 15. Best-first Graph Search
  - ▶ 16. Greedy Best-first Search, A\*, Weighted A\*
  - ▶ 17. IDA\*
  - ▶ 18. Properties of A\*, Part I
  - ▶ 19. Properties of A\*, Part II

## 19.1 Introduction

## Optimality of A\* without Reopening

We now study A\* **without reopening**.

- ▶ For A\* without reopening, admissibility and consistency together guarantee optimality.
- ▶ We prove this on the following slides, again beginning with a basic lemma.
- ▶ Either of the two properties on its own would **not** be sufficient for optimality. (How would one prove this?)

## Reminder: A\* without Reopening

reminder: A\* without reopening

A\* without Reopening

```

open := new MinHeap ordered by ⟨f, h⟩
if h(init()) < ∞:
    open.insert(make_root_node())
closed := new HashSet
while not open.is_empty():
    n := open.pop_min()
    if n.state ∉ closed:
        closed.insert(n)
        if is_goal(n.state):
            return extract_path(n)
        for each ⟨a, s'⟩ ∈ succ(n.state):
            if h(s') < ∞:
                n' := make_node(n, a, s')
                open.insert(n')
return unsolvable
  
```

## 19.2 Monotonicity Lemma

### A\*: Monotonicity Lemma (1)

Lemma (monotonicity of A\* with consistent heuristics)

Consider A\* with a **consistent** heuristic.

Then:

- 1 If  $n'$  is a child node of  $n$ , then  $f(n') \geq f(n)$ .
- 2 On all paths generated by A\*,  $f$  values are non-decreasing.
- 3 The sequence of  $f$  values of the nodes expanded by A\* is non-decreasing.

German: Monotonielemma

## A\*: Monotonicity Lemma (2)

Proof.

on 1.:

Let  $n'$  be a child node of  $n$  via action  $a$ .Let  $s = n.state$ ,  $s' = n'.state$ .

- ▶ by definition of  $f$ :  $f(n) = g(n) + h(s)$ ,  $f(n') = g(n') + h(s')$
  - ▶ by definition of  $g$ :  $g(n') = g(n) + cost(a)$
  - ▶ by consistency of  $h$ :  $h(s) \leq cost(a) + h(s')$
- $$\rightsquigarrow f(n) = g(n) + h(s) \leq g(n) + cost(a) + h(s')$$
- $$= g(n') + h(s') = f(n')$$

on 2.: follows directly from 1. ...

## A\*: Monotonicity Lemma (3)

Proof (continued).

on 3:

- ▶ Let  $f_b$  be the minimal  $f$  value in *open* at the beginning of a **while** loop iteration in A\*.  
Let  $n$  be the removed node with  $f(n) = f_b$ .
- ▶ to show: at the end of the iteration the minimal  $f$  value in *open* is at least  $f_b$ .
- ▶ We must consider the operations modifying *open*: *open.pop\_min* and *open.insert*.
- ▶ *open.pop\_min* can never decrease the minimal  $f$  value in *open* (only potentially increase it).
- ▶ The nodes  $n'$  added with *open.insert* are children of  $n$  and hence satisfy  $f(n') \geq f(n) = f_b$  according to part 1. □

## 19.3 Optimality of A\* without Reopening

## Optimality of A\* without Reopening

Theorem (optimality of A\* without reopening)

A\* without reopening is optimal when using an *admissible* and *consistent* heuristic.

Proof.

From the monotonicity lemma, the sequence of  $f$  values of nodes removed from the open list is non-decreasing.

- ↪ If multiple nodes with the same state  $s$  are removed from the open list, then their  $g$  values are non-decreasing.
- ↪ If we allowed reopening, it would never happen.
- ↪ With consistent heuristics, A\* without reopening behaves the same way as A\* with reopening.

The result follows because A\* with reopening and admissible heuristics is optimal. □

## 19.4 Time Complexity of A\*

### Time Complexity of A\* (1)

What is the time complexity of A\*?

- ▶ depends strongly on the quality of the heuristic
- ▶ **an extreme case:**  $h = 0$  for all states
  - ↪ A\* identical to uniform cost search
- ▶ **another extreme case:**  $h = h^*$  and  $cost(a) > 0$  for all actions  $a$ 
  - ↪ A\* only expands nodes along an optimal solution
  - ↪  $O(\ell^*)$  expanded nodes,  $O(\ell^* b)$  generated nodes, where
    - ▶  $\ell^*$ : length of the found optimal solution
    - ▶  $b$ : branching factor

### Time Complexity of A\* (2)

more precise analysis:

- ▶ dependency of the runtime of A\* on **heuristic error**

example:

- ▶ unit cost problems with
- ▶ **constant branching factor** and
- ▶ **constant absolute error:**  $|h^*(s) - h(s)| \leq c$  for all  $s \in S$

time complexity:

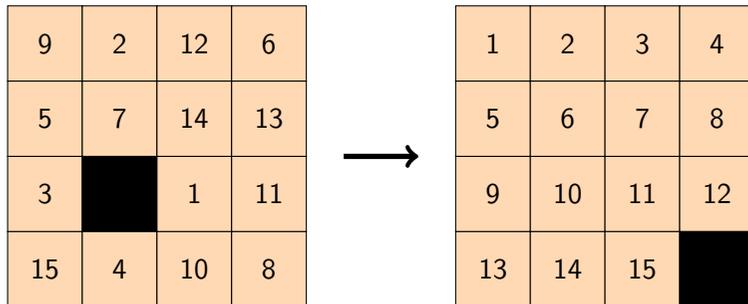
- ▶ **if state space is a tree:** time complexity of A\* grows linearly in solution length (Pohl 1969; Gaschnig 1977)
- ▶ **general search spaces:** runtime of A\* grows exponentially in solution length (Helmert & Röger 2008)

### Overhead of Reopening

How does reopening affect runtime?

- ▶ For most practical state spaces and inconsistent admissible heuristics, the number of reopened nodes is **negligible**.
- ▶ **exceptions** exist:
  - Martelli (1977) constructed state spaces with  $n$  states where **exponentially** many (in  $n$ ) node reopenings occur in A\*.
  - (↪ exponentially worse than uniform cost search)

## Practical Evaluation of A\* (1)



$h_1$ : number of tiles in wrong cell (**misplaced tiles**)

$h_2$ : sum of distances of tiles to their goal cell (**Manhattan distance**)

## Practical Evaluation of A\* (2)

- ▶ experiments with random initial states, generated by **random walk** from goal state
- ▶ entries show **median** of number of **generated nodes** for 101 random walks of the same length  $N$

$N$	generated nodes		
	BFS-Graph	A* with $h_1$	A* with $h_2$
10	63	15	15
20	1,052	28	27
30	7,546	77	42
40	72,768	227	64
50	359,298	422	83
60	> 1,000,000	7,100	307
70	> 1,000,000	12,769	377
80	> 1,000,000	62,583	849
90	> 1,000,000	162,035	1,522
100	> 1,000,000	690,497	4,964

## 19.5 Summary

## Summary

- ▶ **A\* without reopening** using an **admissible and consistent** heuristic is optimal
- ▶ key property **monotonicity lemma** (with consistent heuristics):
  - ▶  $f$  values never decrease along paths considered by A\*
  - ▶ sequence of  $f$  values of expanded nodes is non-decreasing
- ▶ time complexity depends on heuristic and shape of state space
  - ▶ precise details complex and depend on many aspects
  - ▶ reopening increases runtime exponentially in degenerate cases, but usually negligible overhead
  - ▶ small improvements in heuristic values often lead to exponential improvements in runtime