Foundations of Artificial Intelligence

B1. State-Space Search: State Spaces

Malte Helmert

University of Basel

February 24, 2025

B1.1 State-Space Search Problems

B1.2 Formalization

B1.3 State-Space Search

B1.4 Summary

State-Space Search: Overview

Chapter overview: state-space search

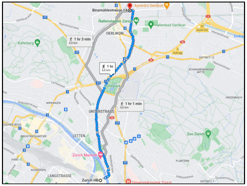
- B1–B3. Foundations
  - B1. State Spaces
  - B2. Representation of State Spaces
  - B3. Examples of State Spaces
- B4–B8. Basic Algorithms
- B9–B15. Heuristic Algorithms

B1.1 State-Space Search Problems

State-Space Search Applications

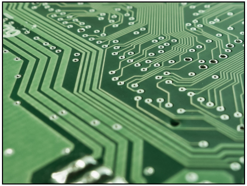
Mario AI competition

route planning



multi-agent path finding

scheduling



software/hardware verification

NPC behaviour

Classical Assumptions

"classical" assumptions considered in this part of the course:

- no other agents in the environment (single-agent)
- always knows state of the world (fully observable)
- state only changed by the agent (static)
- finite number of states/actions (in particular discrete)
- actions have deterministic effect on the state

⇝ can all be generalized (but not in this part of the course)

Classification

classification:

State-Space Search

environment:

- static vs. dynamic
- deterministic vs. nondeterministic vs. stochastic
- fully observable vs. partially observable
- discrete vs. continuous
- single-agent vs. multi-agent

problem solving method:

- problem-specific vs. general vs. learning

Informal Description

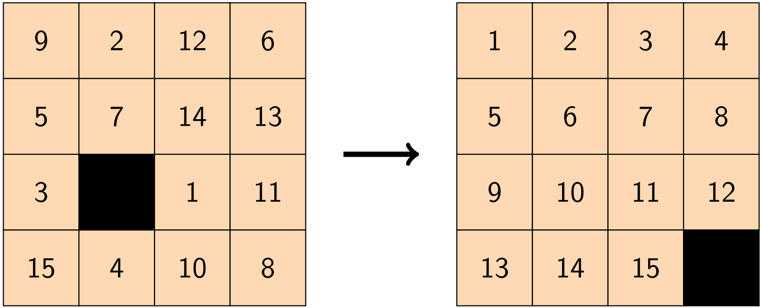
State-space search problems are among the "simplest" and most important classes of AI problems.

objective of the agent:

- apply a sequence of actions
- that reaches a goal state
- from a given initial state

performance measure: minimize total action cost

Motivating Example: 15-Puzzle



B1.2 Formalization

State Spaces

Definition (state space)

A state space or transition system is a 6-tuple $\mathcal{S} = \langle S, A, cost, T, s_\mathrm{I}, S_\mathrm{G} \rangle$ with

- finite set of states S
- finite set of actions A
- action costs $cost : A \to \mathbb{R}_0^+$
- transition relation $T \subseteq S \times A \times S$ that is deterministic in $\langle s, a \rangle$ (see next slide)
- initial state $s_\mathrm{I} \in S$
- set of goal states $S_\mathrm{G} \subseteq S$

German: Zustandsraum, Transitionssystem, Zustände, Aktionen, Aktionskosten, Transitions-/Übergangsrelation, deterministisch, Anfangszustand, Zielzustände

State Spaces: Terminology & Notation

Definition (transition, deterministic)

Let $\mathcal{S} = \langle S, A, cost, T, s_\mathrm{I}, S_\mathrm{G} \rangle$ be a state space.

The triples $\langle s, a, s' \rangle \in T$ are called (state) transitions.

We say $\mathcal{S}$ has the transition $\langle s, a, s' \rangle$ if $\langle s, a, s' \rangle \in T$.
We write this as $s \xrightarrow{a} s'$, or $s \rightarrow s'$ when a does not matter.

Transitions are deterministic in $\langle s, a \rangle$: it is forbidden to have both $s \xrightarrow{a} s_1$ and $s \xrightarrow{a} s_2$ with $s_1 \neq s_2$.

State Space: Running Example

Consider the bounded inc-and-square search problem.

informal description:

- find a sequence of
  - increment-mod10 (*inc*) and
  - square-mod10 (*sqr*) actions
- on the natural numbers from 0 to 9
- that reaches the number 6 or 7
- starting from the number 1
- assuming each action costs 1.

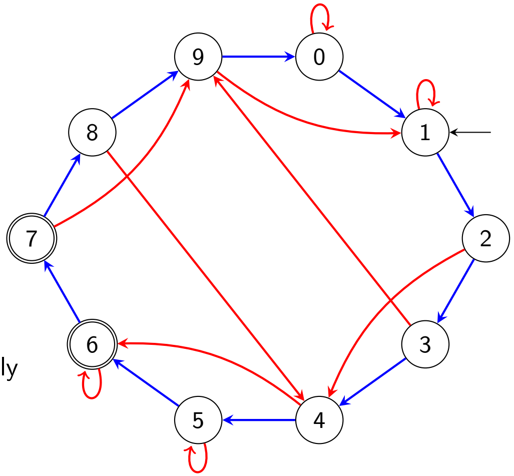
formal model:

- $S = \{0, 1, \dots, 9\}$
- $A = \{inc, sqr\}$
- $cost(inc) = cost(sqr) = 1$
- T s.t. for $i = 0, \dots, 9$:
  - $\langle i, inc, (i + 1) \bmod 10 \rangle \in T$
  - $\langle i, sqr, i^2 \bmod 10 \rangle \in T$
- $s_I = 1$
- $S_G = \{6, 7\}$

Graph Interpretation

state spaces are often depicted as directed, labeled graphs

- states: graph vertices
- transitions: labeled arcs (here: colors instead of labels)
- initial state: incoming arrow
- goal states: double circles
- actions: the arc labels
- action costs: described separately (or implicitly $= 1$)



State Spaces: More Terminology (1)

We use common terminology from graph theory.

Definition (predecessor, successor, applicable action)

Let $\mathcal{S} = \langle S, A, cost, T, s_\mathrm{I}, S_\mathrm{G} \rangle$ be a state space.

Let $s, s' \in S$ be states with $s \to s'$.

- s is a **predecessor** of s'
- s' is a **successor** of s

If $s \xrightarrow{a} s'$, then action a is **applicable** in s.

German: Vorgänger, Nachfolger, anwendbar

State Spaces: More Terminology (2)

Definition (path)

Let $\mathcal{S} = \langle S, A, cost, T, s_\mathrm{I}, S_\mathrm{G}\rangle$ be a state space.

Let $s_0, \dots, s_n \in S$ be states and $a_1, \dots, a_n \in A$ be actions such that $s_0 \xrightarrow{a_1} s_1, \dots, s_{(n-1)} \xrightarrow{a_n} s_n$.

- $\pi = \langle a_1, \dots, a_n\rangle$ is a path from s_0 to s_n
- length of π: $|\pi| = n$
- cost of π: $cost(\pi) = \sum_{i=1}^{n} cost(a_i)$

German: Pfad, Länge, Kosten

- paths may have length 0
- sometimes "path" is used for state sequence $\langle s_0, \dots, s_n\rangle$ or sequence $\langle s_0, a_1, s_1, \dots, s_{(n-1)}, a_n, s_n\rangle$

State Spaces: More Terminology (3)

More terminology:

Definition (reachable, solution, optimal)

Let $\mathcal{S} = \langle S, A, cost, T, s_I, S_G \rangle$ be a state space.

- state s is **reachable** if a path from s_I to s exists
- paths from $s \in S$ to some state $s_G \in S_G$ are **solutions for/from s**
- solutions for s_I are called **solutions for $\mathcal{S}$**
- **optimal solutions** (for s) have minimal costs among all solutions (for s)

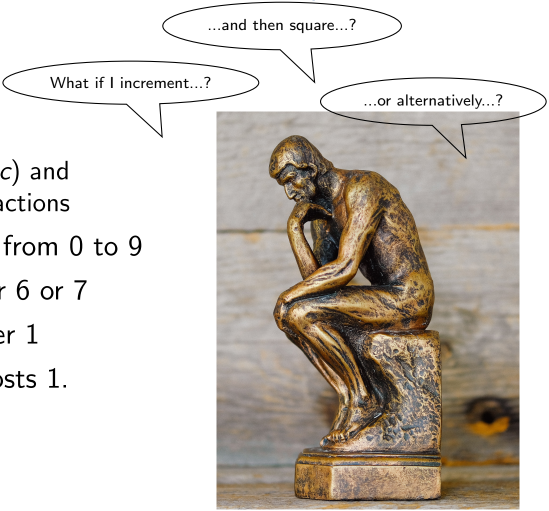
German: erreichbar, Lösung für/von s, optimale Lösung

B1.3 State-Space Search

Solving Search Problems

Consider again the running example.

How do you solve this?



informal description:

- find a sequence of
  - increment-mod10 (*inc*) and
  - square-mod10 (*sqr*) actions
- on the natural numbers from 0 to 9
- that reaches the number 6 or 7
- starting from the number 1
- assuming each action costs 1.

State-Space Search

State-Space Search

State-space search is the algorithmic problem
of finding solutions in state spaces
or proving that no solution exists.

In optimal state-space search, only optimal solutions
may be returned.

German: Zustandsraumsuche, optimale Zustandsraumsuche

Learning Objectives for State-Space Search

Learning Objectives for the Topic of State-Space Search

- understanding state-space search:
  What is the problem and how can we formalize it?
- evaluate search algorithms:
  completeness, optimality, time/space complexity
- get to know search algorithms:
  uninformed vs. informed; tree and graph search
- evaluate heuristics for search algorithms:
  goal-awareness, safety, admissibility, consistency
- efficient implementation of search algorithms
- experimental evaluation of search algorithms
- design and comparison of heuristics for search algorithms

B1.4 Summary

Summary

- state-space search problems:
  find action sequence leading from initial state to a goal state
- performance measure: sum of action costs
- formalization via state spaces:
  - states, actions, action costs, transitions,
    initial state, goal states
- terminology for transitions, paths, solutions
- definition of (optimal) state-space search