Artificial Intelligence Planning

Introduction and Overview

Overview

- What is Planning (in AI)?
- A Conceptual Model for Planning
- Planning and Search
- Example Problems

Human Planning and Acting

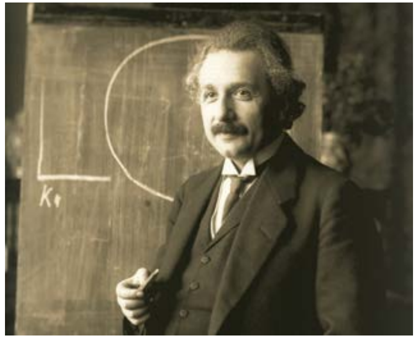
- acting without (explicit) planning:
  - when purpose is immediate
  - when performing well-trained behaviours
  - when course of action can be freely adapted
- acting after planning:
  - when addressing a new situation
  - when tasks are complex
  - when the environment imposes high risk/cost
  - when collaborating with others

- people plan only when strictly necessary

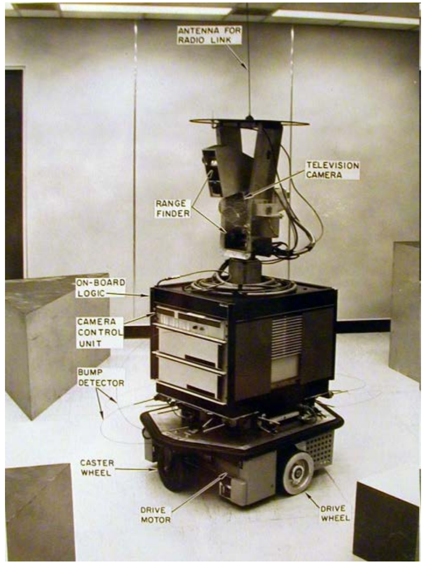
Defining AI Planning

- planning:
  - explicit deliberation process that chooses and organizes actions by anticipating their outcomes
  - aims at achieving some pre-stated objectives
- AI planning:
  - computational study of this deliberation process

Why Study Planning in AI?



- scientific goal of AI:
  understand intelligence
  - planning is an important component of rational (intelligent) behaviour



- engineering goal of AI:
  build intelligent entities
  - build planning software for choosing and organizing actions for autonomous intelligent machines

Domain-Specific vs. Domain-Independent Planning

- domain-specific planning: use specific representations and techniques adapted to each problem
  - important domains: path and motion planning, perception planning, manipulation planning, communication planning
- domain-independent planning: use generic representations and techniques
  - exploit commonalities to all forms of planning
  - leads to general understanding of planning

- domain-independent planning complements domain-specific planning

Quiz: True or False?

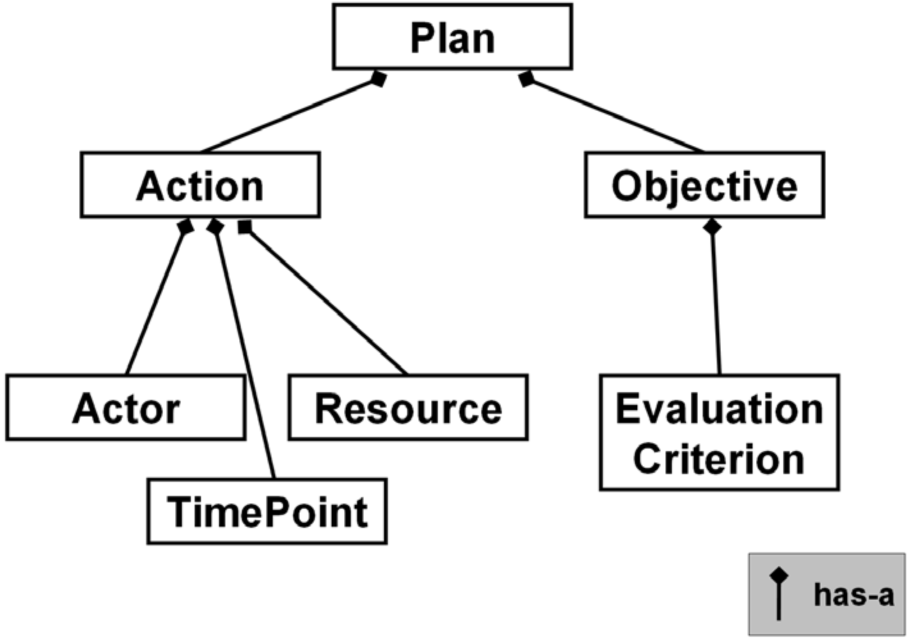
- people only plan *when they have to* because the benefit of an optimal plan does not always justify the effort of planning
- for humans, planning is a *subconscious process*, which is why computational planning is so hard
- planning involves a *mental simulation of actions* to foresee future world states and compare them to goals
- in Artificial Intelligence, planning is concerned with the search for *computationally optimal* plans
- domain-specific planning is used when efficiency is vital, whereas domain-independent planning is good for planning from first principles

Overview

- What is Planning (in AI)?
- A Conceptual Model for Planning
- Planning and Search
- Example Problems

Why a Conceptual Model?

- conceptual model: theoretical device for describing the elements of a problem
- good for:
  - explaining basic concepts
  - clarifying assumptions
  - analyzing requirements
  - proving semantic properties
- not good for:
  - efficient algorithms and computational concerns



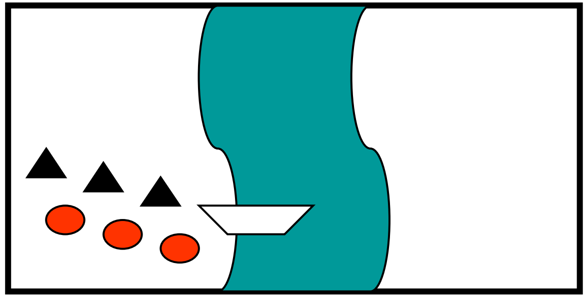
Conceptual Model for Planning: State-Transition Systems

- A <u>state-transition system</u> is a 4-tuple $\Sigma = (S,A,E,\gamma)$, where:
  - $S = \{s_1,s_2,\ldots\}$ is a finite or recursively enumerable set of states;
  - $A = \{a_1,a_2,\ldots\}$ is a finite or recursively enumerable set of actions;
  - $E = \{e_1,e_2,\ldots\}$ is a finite or recursively enumerable set of events; and
  - γ: $S \times (A \cup E) \rightarrow 2^S$ is a state transition function.
- if $a \in A$ and $\gamma(s,a) \neq \emptyset$ then a is <u>applicable</u> in s
- applying a in s will take the system to $s' \in \gamma(s,a)$

State-Transition Systems as Graphs

- A state-transition system $\Sigma = (S,A,E,\gamma)$ can be represented by a directed labelled graph $G = (N_G,E_G)$ where:
  - the nodes correspond to the states in S, i.e. $N_G=S$; and
  - there is an arc from $s \in N_G$ to $s' \in N_G$, i.e. $s \rightarrow s' \in E_G$, with label $u \in (A \cup E)$ if and only if $s' \in \gamma(s,u)$.

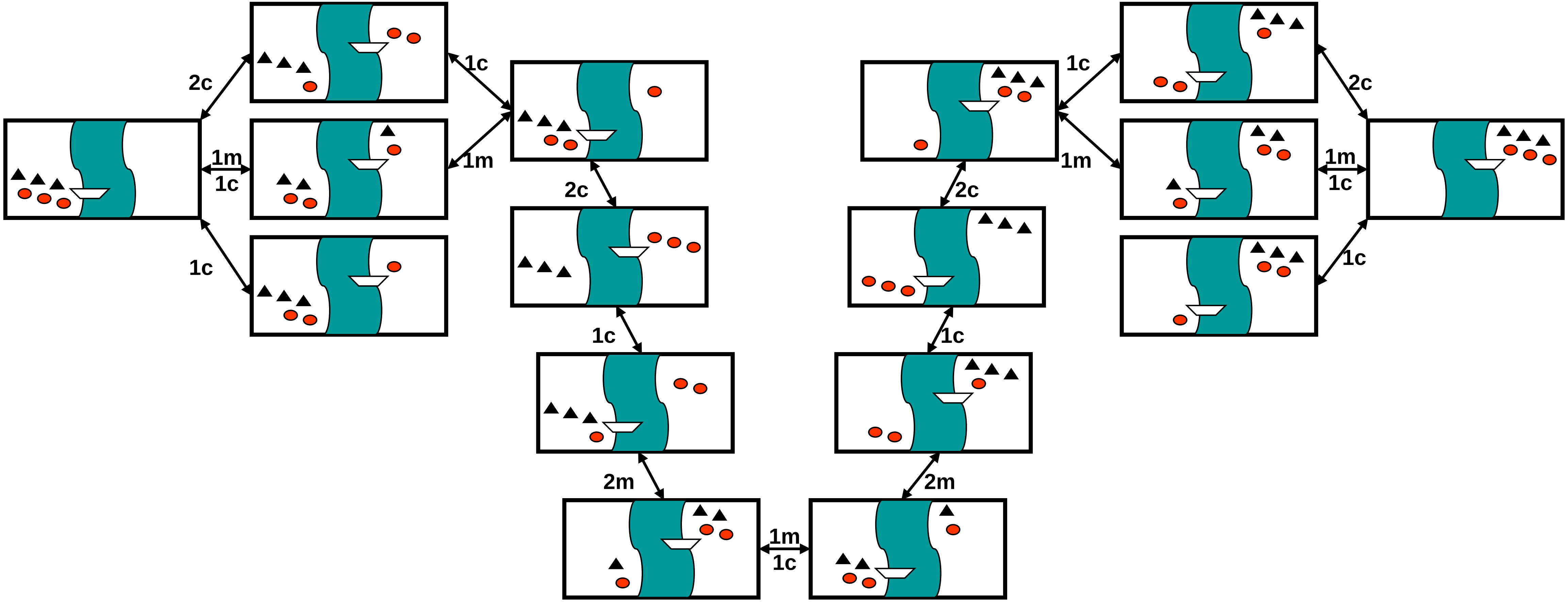
Toy Problem: Missionaries and Cannibals



On one bank of a river are three missionaries (black triangles) and three cannibals (red circles). There is one boat available that can hold up to two people and that they would like to use to cross the river. If the cannibals ever outnumber the missionaries on either of the river's banks, the missionaries will get eaten.

How can the boat be used to safely carry all the missionaries and cannibals across the river?

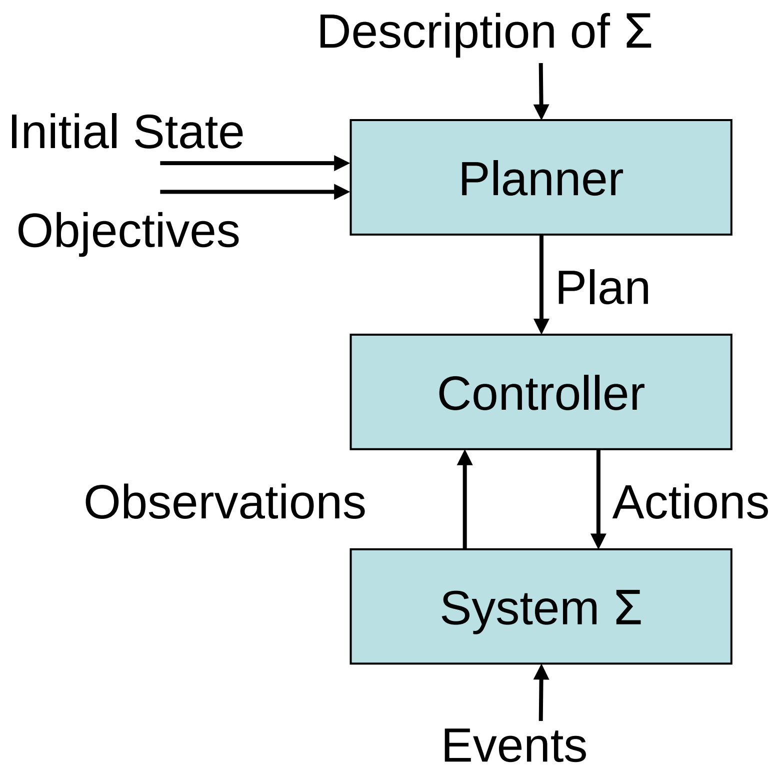
State-Transition Graph Example: Missionaries and Cannibals



Objectives and Plans

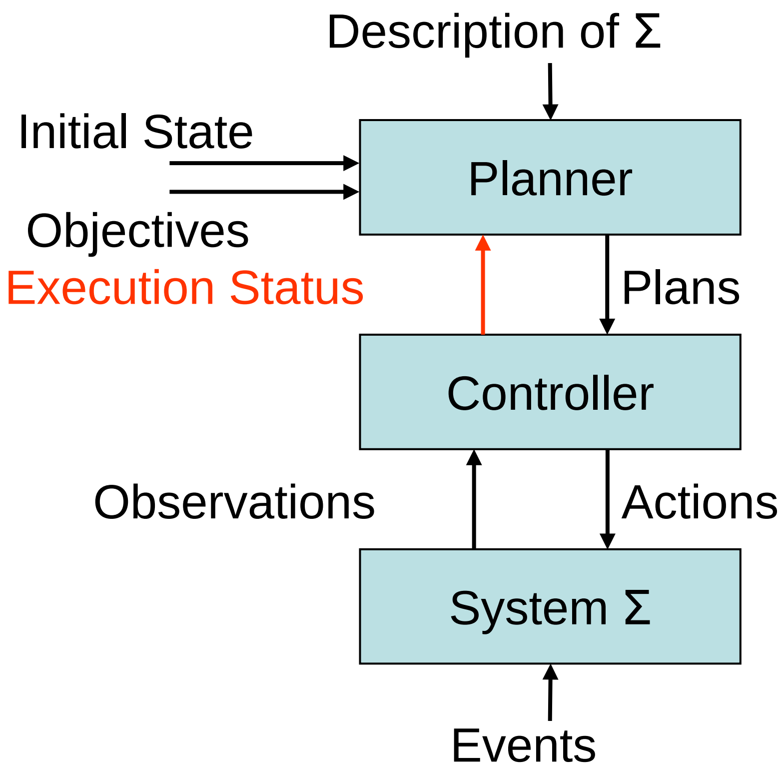
- state-transition system:
  - describes all ways in which a system may evolve
- plan:
  - a structure that gives appropriate actions to apply in order to achieve some objective when starting from a given state
- types of objective:
  - goal state s_g or set of goal states S_g
  - satisfy some conditions over the sequence of states
  - optimize utility function attached to states
  - task to be performed

Planning and Plan Execution



- planner:
  - given: description of Σ, initial state, objective
  - generate: plan that achieves objective
- controller:
  - given: plan, current state (observation function: $\eta: S \rightarrow O$)
  - generate: action
- state-transition system:
  - evolves as actions are executed and events occur

Dynamic Planning



- problem: real world differs from model described by Σ
- more realistic model: interleaved planning and execution
  - plan supervision
  - plan revision
  - re-planning
- dynamic planning: closed loop between planner and controller
  - execution status

Overview

- What is Planning (in AI)?
- A Conceptual Model for Planning
- Planning and Search
- Example Problems

Toy Problems vs. Real-World Problems

Toy Problems/Puzzles

- concise and exact description
- used for illustration purposes (e.g. here)
- used for performance comparisons

Real-World Problems

- no single, agreed-upon description
- people care about the solutions

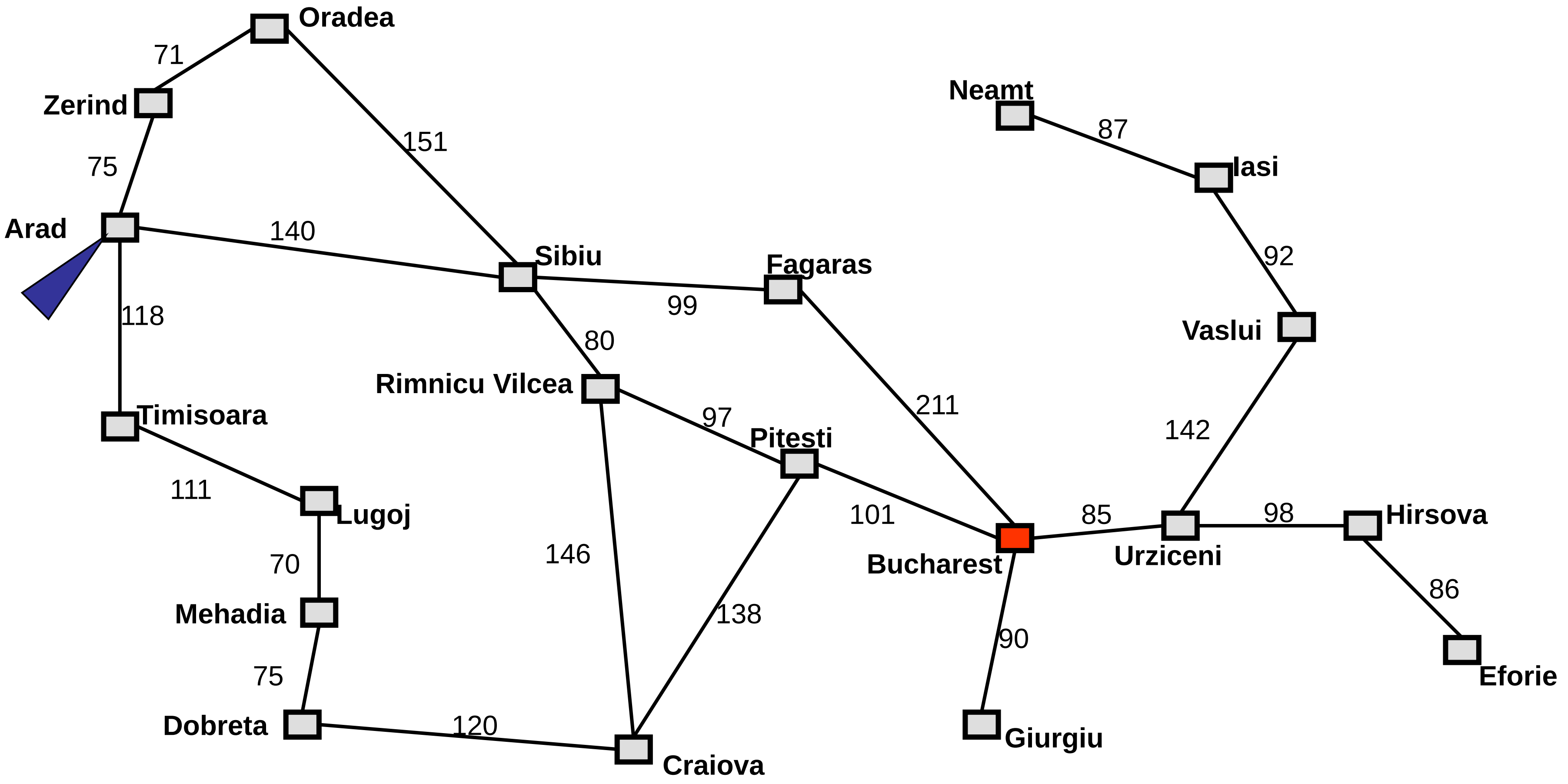
Search Problems

- <u>initial state</u>
- set of possible <u>actions</u>/applicability conditions
  - successor function: *state* → set of <*action*, *state*>
  - successor function + initial state = state space
  - path (solution)
- <u>goal</u>
  - goal state or goal test function
- <u>path cost</u> function
  - for optimality
  - assumption: path cost = sum of step costs

Missionaries and Cannibals: Successor Function

state	set of <*action*, *state*>
(L:3m,3c,b-R:0m,0c) →	{<2c, (L:3m,1c-R:0m,2c,b)>, <1m1c, (L:2m,2c-R:1m,1c,b)>, <1c, (L:3m,2c-R:0m,1c,b)>}
(L:3m,1c-R:0m,2c,b) →	{<2c, (L:3m,3c,b-R:0m,0c)>, <1c, (L:3m,2c,b-R:0m,1c)>}
(L:2m,2c-R:1m,1c,b) →	{<1m1c, (L:3m,3c,b-R:0m,0c)>, <1m, (L:3m,2c,b-R:0m,1c)>}
⋮	⋮

Real-World Problem: Touring in Romania



Problem Formulation

- problem formulation
  - process of deciding what actions and states to consider
  - granularity/abstraction level

- assumptions about the environment:
  - finite and discrete
  - fully observable
  - deterministic
  - static

- other assumptions:
  - restricted goals
  - sequential plans
  - implicit time
  - offline planning

Search Nodes

- <u>search nodes</u>: the nodes in the search tree
- data structure:
  - *state*: a state in the state space
  - *parent node*: the immediate predecessor in the search tree
  - *action*: the action that, performed in the parent node's state, leads to this node's state
  - *path cost*: the total cost of the path leading to this node
  - *depth*: the depth of this node in the search tree

General Tree Search Algorithm

function treeSearch(*problem*, *strategy*)

    fringe ← { **new** searchNode(*problem*.initialState) }

    loop

        if empty(*fringe*) **then return** failure

        node ← selectFrom(*fringe*, *strategy*)

        if *problem*.goalTest(*node.state*) **then**

            return pathTo(*node*)

        fringe ← *fringe* + expand(*problem*, *node*)

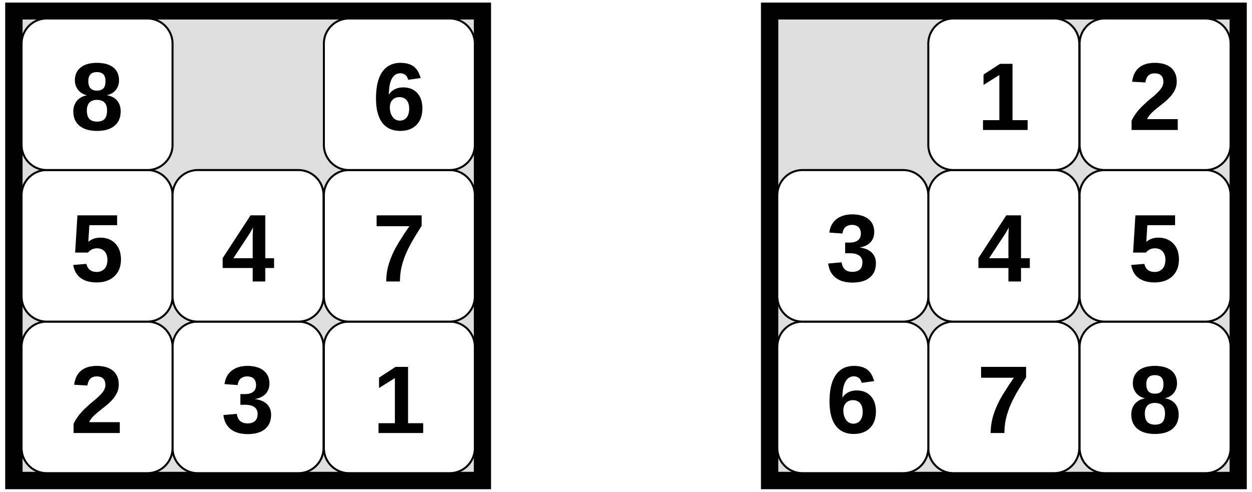
Search (Control) Strategy

- search or control strategy: an effective method for scheduling the application of the successor function to expand nodes
  - selects the next node to be expanded from the fringe
  - determines the order in which nodes are expanded
  - aim: produce a goal state as quickly as possible
- examples:
  - LIFO/FIFO-queue for fringe nodes
  - alphabetical ordering

Overview

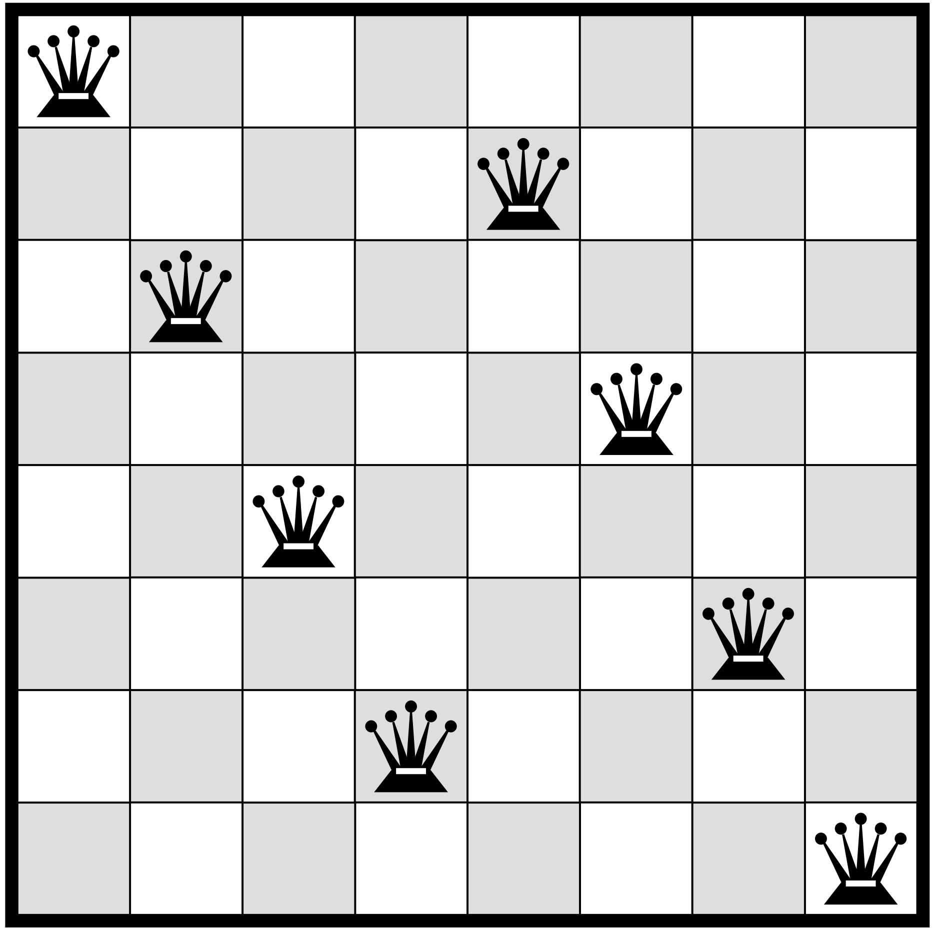
- What is Planning (in AI)?
- A Conceptual Model for Planning
- Planning and Search
- Example Problems

Toy Problem:
Sliding-Block Puzzle



The 8-Puzzle

Toy Problem: *N*-Queens Problem

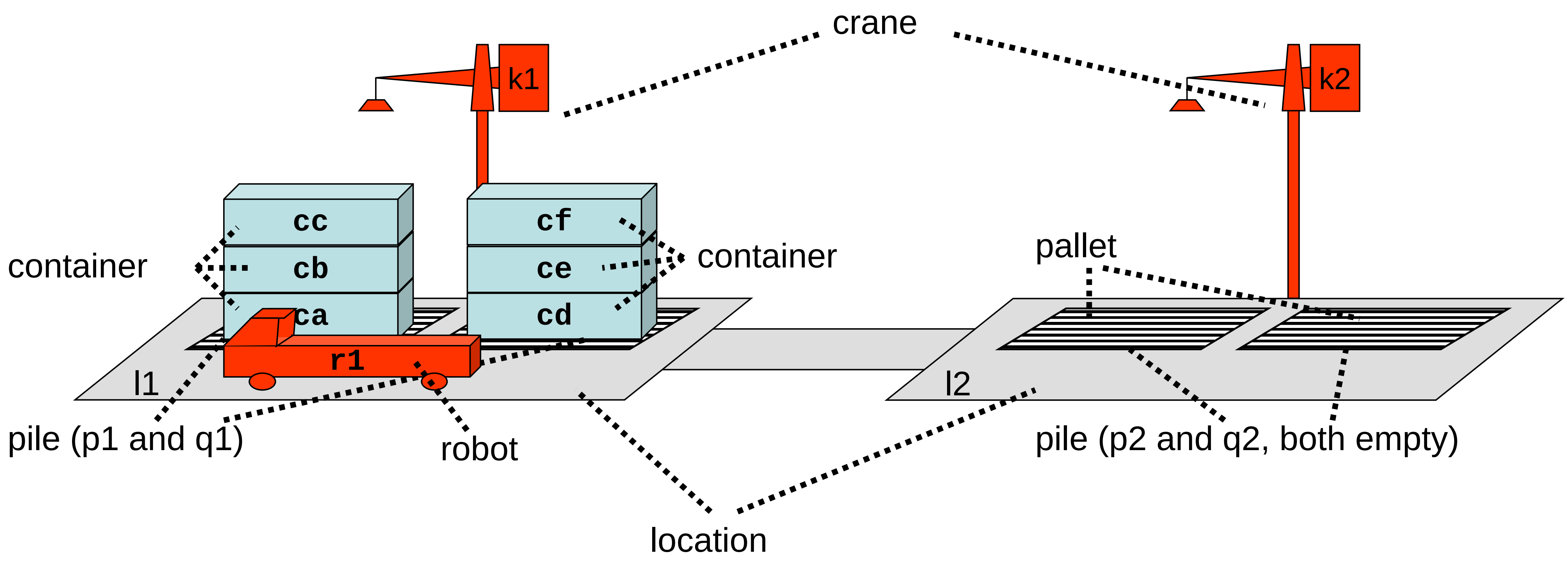


Place *n* queens on an *n* by *n* chess board such that none of the queens attacks any of the others.

The Dock-Worker Robots (DWR) Domain

- aim: have one example to illustrate planning procedures and techniques
- informal description:
  - harbour with several locations (docks), docked ships, storage areas for containers, and parking areas for trucks and trains
  - cranes to load and unload ships etc., and robot carts to move containers around

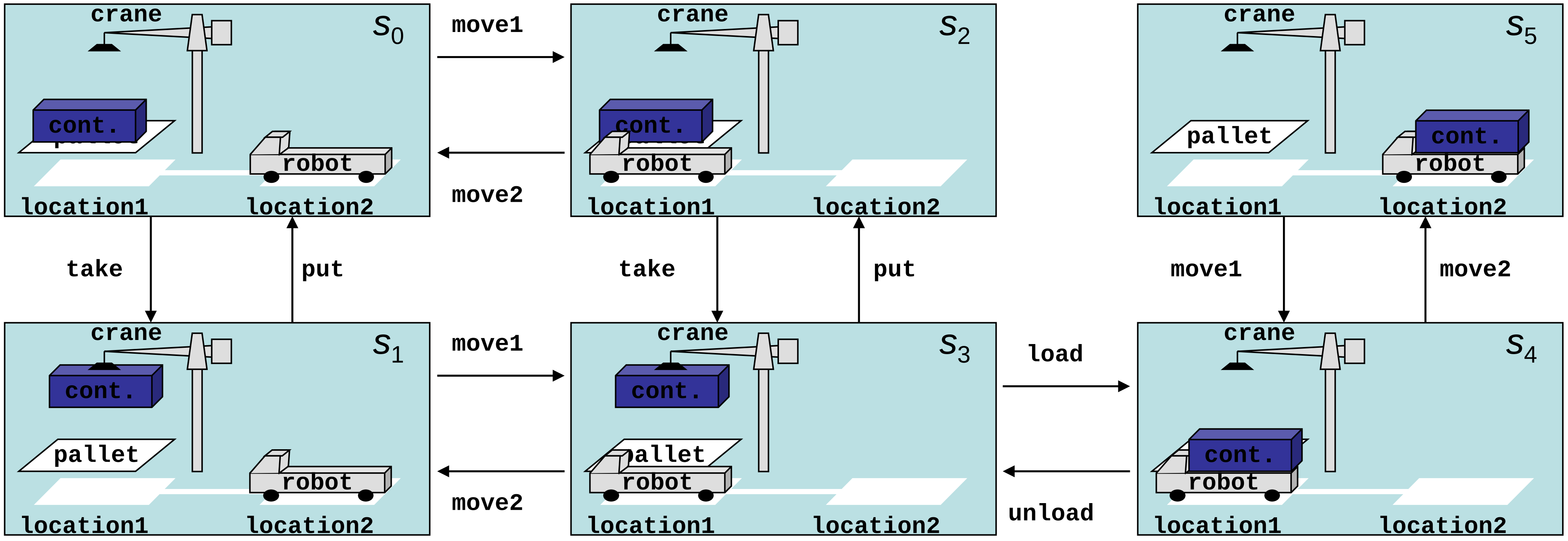
DWR Example State



Actions in the DWR Domain

- **`move`** robot *r* from location *l* to some adjacent and unoccupied location *l'*
- **`take`** container *c* with empty crane *k* from the top of pile *p*, all located at the same location *l*
- **`put`** down container *c* held by crane *k* on top of pile *p*, all located at location *l*
- **`load`** container *c* held by crane *k* onto unloaded robot *r*, all located at location *l*
- **`unload`** container *c* with empty crane *k* from loaded robot *r*, all located at location *l*

State-Transition Systems: Graph Example



Overview

- What is Planning (in AI)?
- A Conceptual Model for Planning
- Planning and Search
- Example Problems