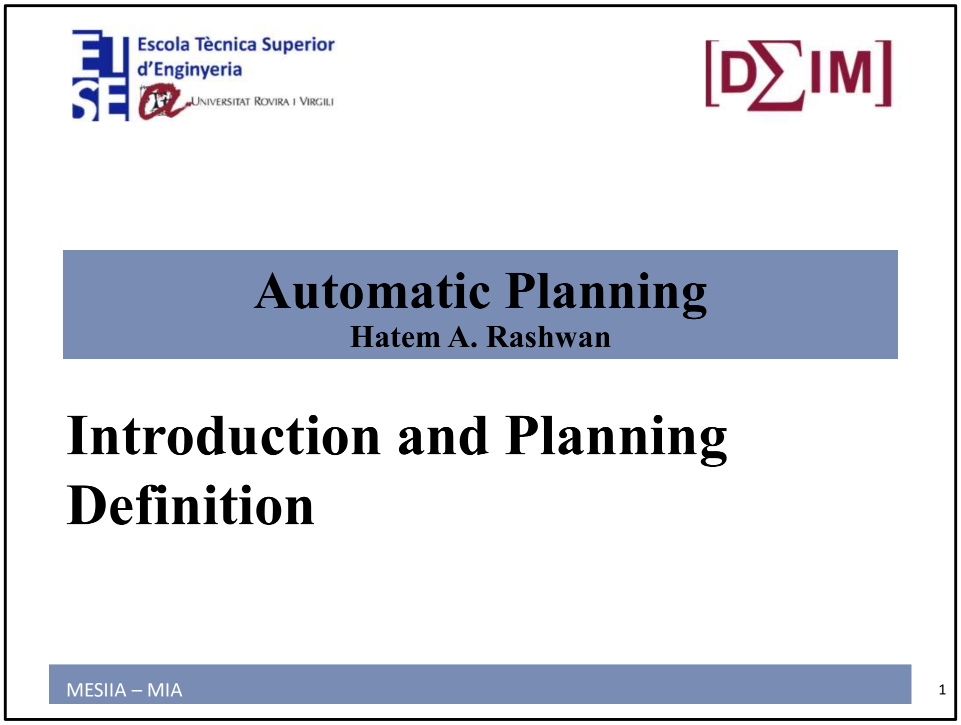


Escola Tècnica Superior d'Enginyeria

Universitat Rovira i Virgili

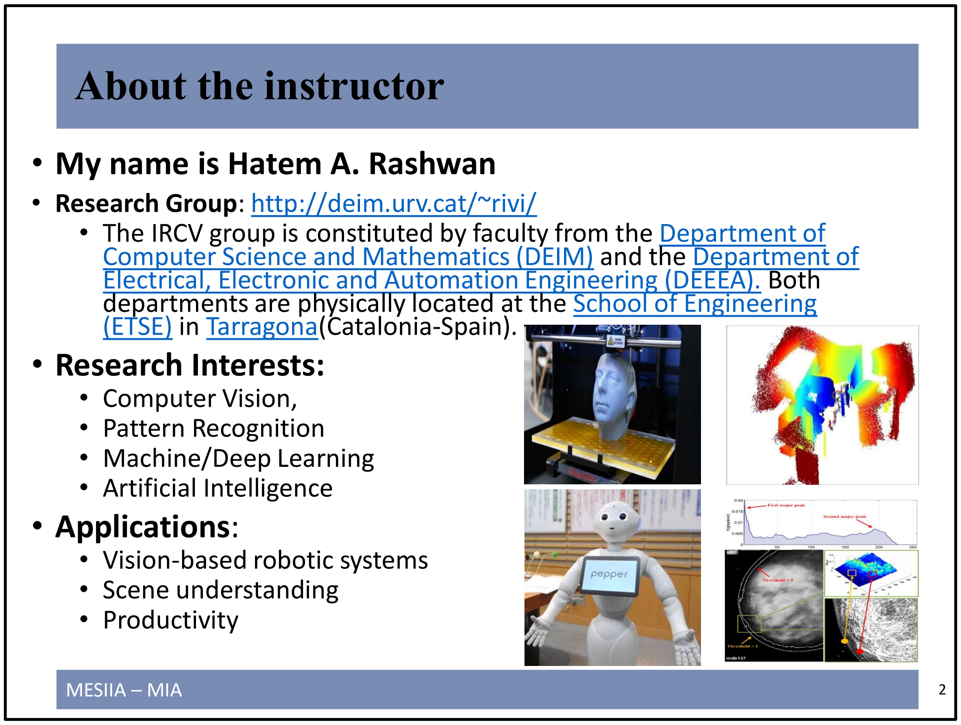
[DΣIM]

Automatic Planning

Hatem A. Rashwan

Introduction and Planning Definition

This lecture will include some examples and introduction to the basic techniques that we will use to solve planning problems.



About the instructor

- **My name is Hatem A. Rashwan**
- **Research Group**: http://deim.urv.cat/~rivi/
  - The IRCV group is constituted by faculty from the Department of Computer Science and Mathematics (DEIM) and the Department of Electrical, Electronic and Automation Engineering (DEEEA). Both departments are physically located at the School of Engineering (ETSE) in Tarragona(Catalonia-Spain).
- **Research Interests:**
  - Computer Vision,
  - Pattern Recognition
  - Machine/Deep Learning
  - Artificial Intelligence
- **Applications**:
  - Vision-based robotic systems
  - Scene understanding
  - Productivity

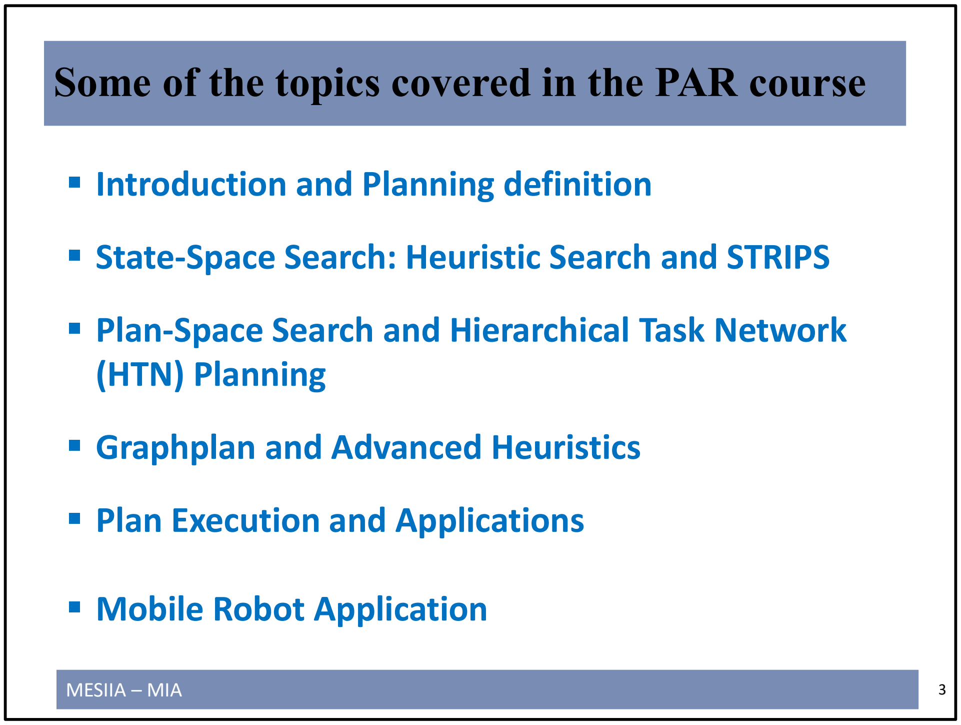
MESIIA – MIA

2

The research group is called IRCV (Intelligence Robotics and Computer Vision).

We are interested at:

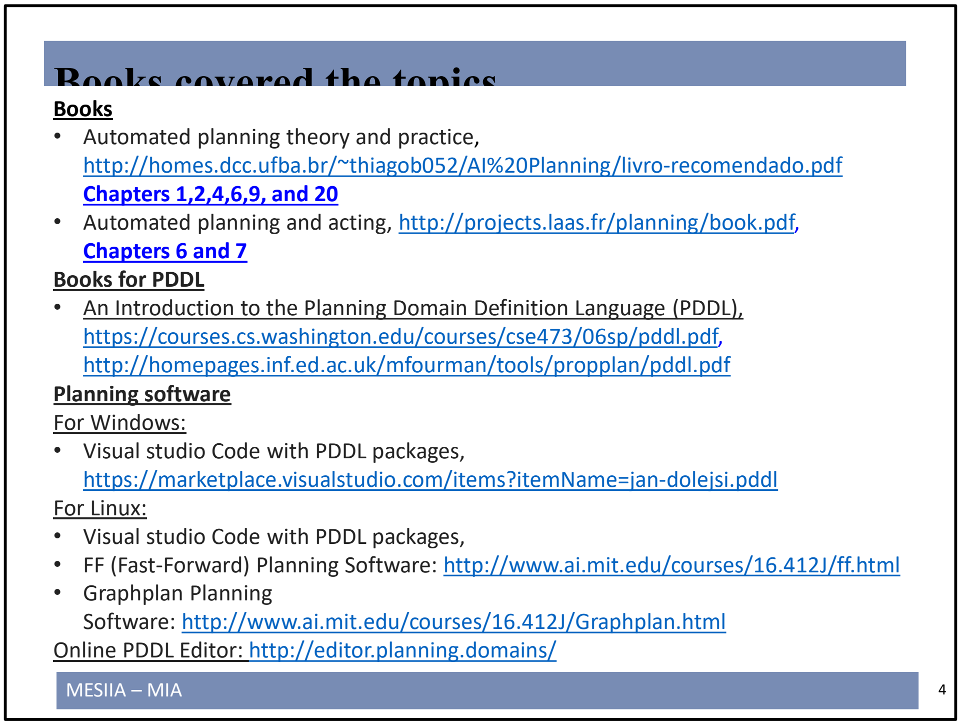
AI in general and mainly at:
Computer Vision,
Pattern Recognition
Machine/Deep Learning



Some of the topics covered in the PAR course

- Introduction and Planning definition
- State-Space Search: Heuristic Search and STRIPS
- Plan-Space Search and Hierarchical Task Network (HTN) Planning
- Graphplan and Advanced Heuristics
- Plan Execution and Applications
- Mobile Robot Application

These the contents of the course during 6 lectures.



Books covered the topics

Books

- Automated planning theory and practice, http://homes.dcc.ufba.br/~thiagob052/AI%20Planning/livro-recomendado.pdf **Chapters 1,2,4,6,9, and 20**
- Automated planning and acting, http://projects.laas.fr/planning/book.pdf, **Chapters 6 and 7**

Books for PDDL

- An Introduction to the Planning Domain Definition Language (PDDL), https://courses.cs.washington.edu/courses/cse473/06sp/pddl.pdf, http://homepages.inf.ed.ac.uk/mfourman/tools/propplan/pddl.pdf

Planning software

For Windows:

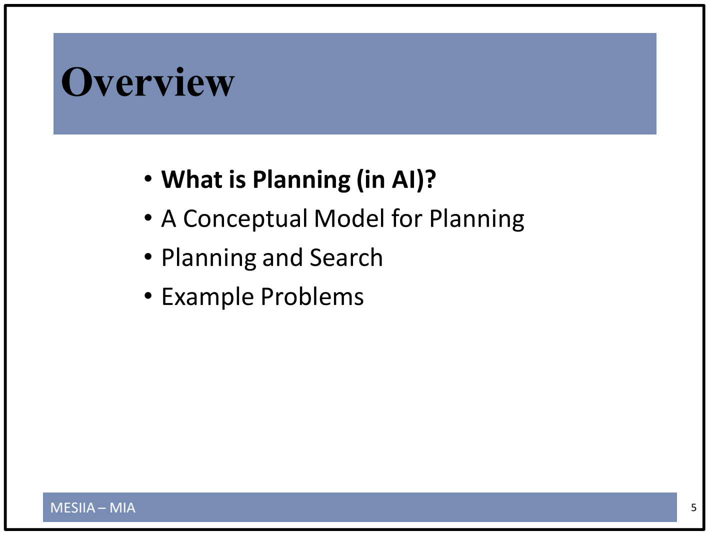
- Visual studio Code with PDDL packages, https://marketplace.visualstudio.com/items?itemName=jan-dolejsi.pddl

For Linux:

- Visual studio Code with PDDL packages,
- FF (Fast-Forward) Planning Software: http://www.ai.mit.edu/courses/16.412J/ff.html
- Graphplan Planning Software: http://www.ai.mit.edu/courses/16.412J/Graphplan.html

Online PDDL Editor: http://editor.planning.domains/

References you can use to follow the planning course and how to use the PDDL language to define different planning problems. Visual studio code can be installed on Windows, Linux and Mac.



Overview

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

MESIIA – MIA

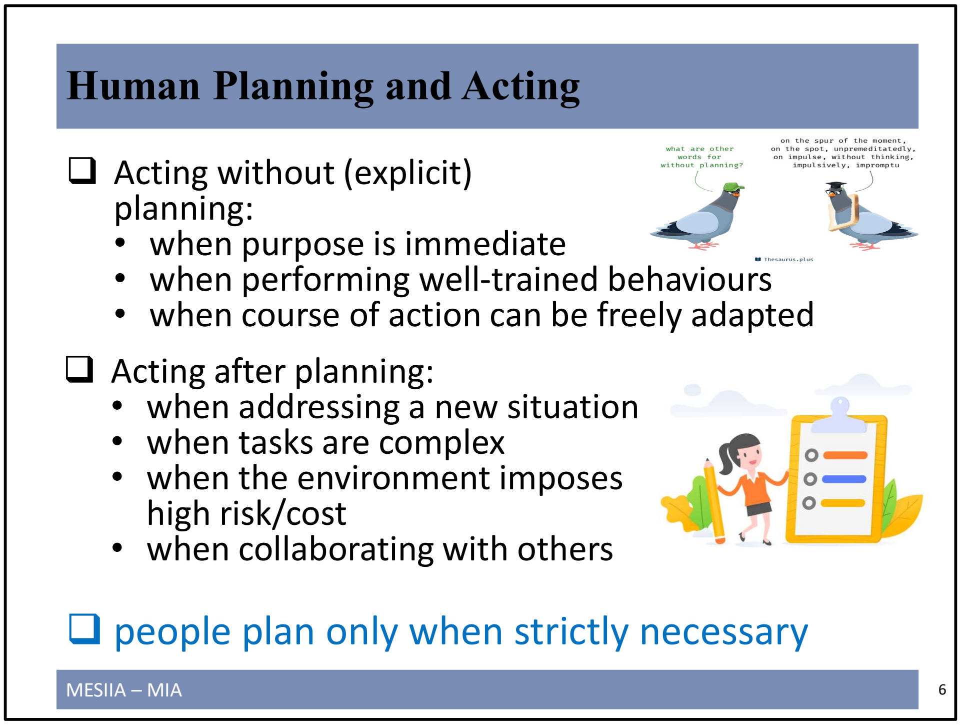
5

Overview

➢ **What is Planning (in AI)?**

•just done: what do we mean by (AI) planning?

- **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

what are other words for without planning?
on the spur of the moment, on the spot, unpremeditatedly, on impulse, without thinking, impulsively, impromptu
Thesaurus.plus

❑ 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

Human Planning and Acting

•humans rarely plan before acting in everyday situations

•**acting without (explicit) planning:** (may be subconscious)

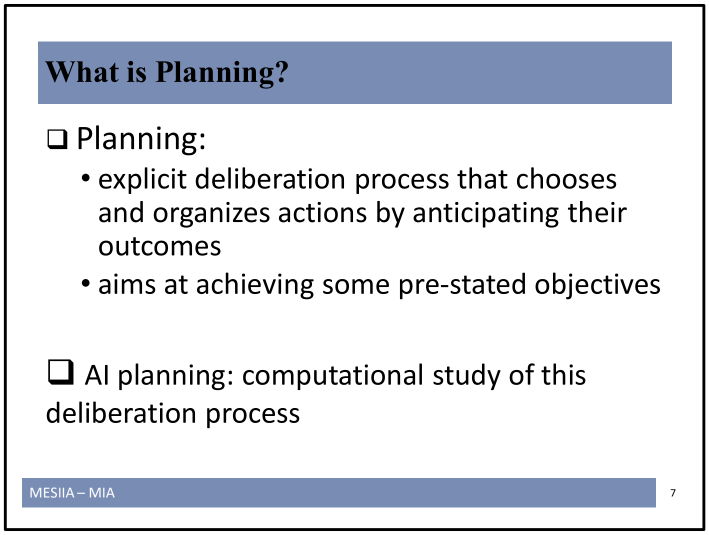
- •**when purpose is immediate** (e.g., when you want to sit down)
- •**when performing well-trained behaviours** (e.g., drive car)
- •**when course of action can be freely adapted** (e.g., shopping)

•**acting after planning:**

- •**when addressing a new situation** (e.g., move to a new house)
- •**when tasks are complex** (e.g., plan to buy a house)
- •**when the environment imposes high risk/cost** (e.g., manage a plant)
- •**when collaborating with others** (e.g. build a hospital or construct a company)

•**people plan only when strictly necessary**

- •because planning is complicated and time-consuming (trade-off: efforts vs. benefit)
- •often, we seek only good rather than optimal plans



What is 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

Defining AI Planning

•**planning**:

•**explicit deliberation process that chooses and organizes actions by anticipating their outcomes**

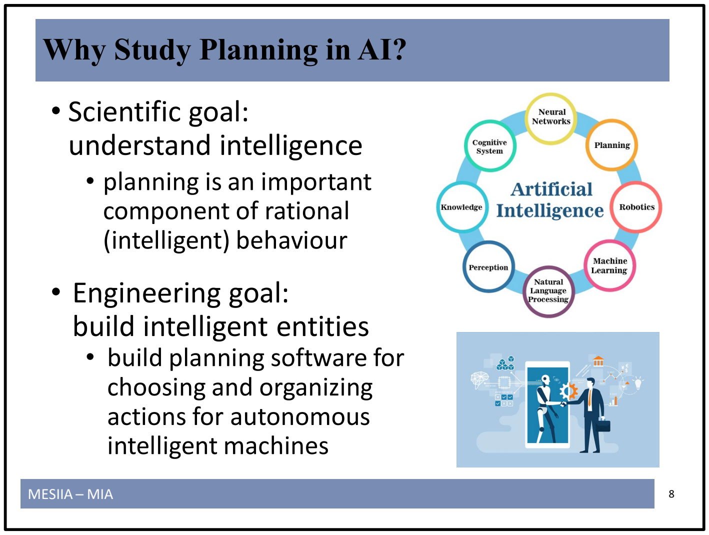
•in short: planning is reasoning about actions, which can already done by the agent and anticipating the outcomes of each action.

•**aims at achieving some pre-stated objectives**

•or: achieving them as best as possible (planning as optimization problem)

•**AI planning**:

•**computational study of this deliberation process** by building a computational model for this planning.



Why Study Planning in AI?

- Scientific goal: understand intelligence
  - planning is an important component of rational (intelligent) behaviour
- Engineering goal: build intelligent entities
  - build planning software for choosing and organizing actions for autonomous intelligent machines

Why Study Planning in AI?

•scientific goal of AI: understand intelligence

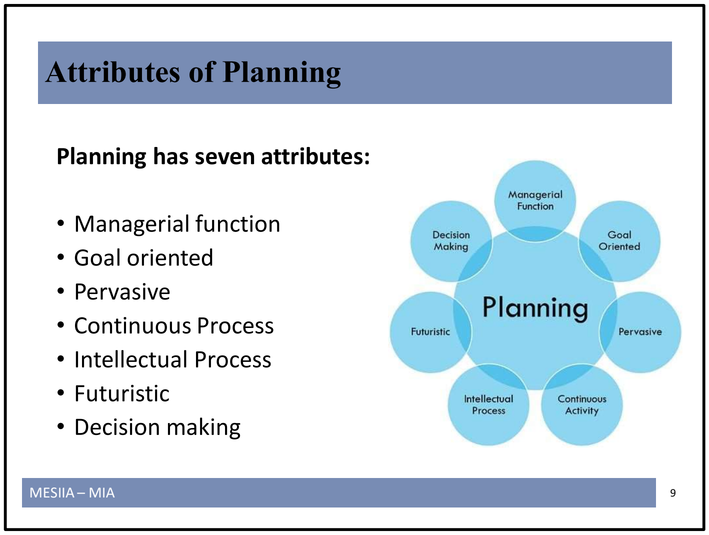
•planning is an important component of intelligent behaviour

•planning is part of intelligent behaviour

•engineering goal of AI: build intelligent entities

•build planning software for choosing and organizing actions for autonomous intelligent machines

•example: Mars explorer (cannot be remotely operated)

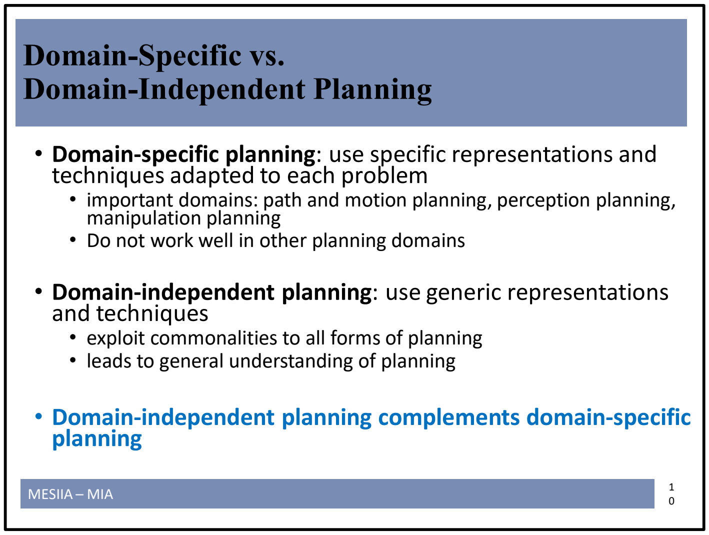


Attributes of Planning

Planning has seven attributes:

- Managerial function
- Goal oriented
- Pervasive
- Continuous Process
- Intellectual Process
- Futuristic
- Decision making

1.Managerial function: Planning is a first and foremost managerial function provides the base for other functions of the management, i.e., organizing, staffing, directing and controlling, as all people/agents are performed within the periphery of the plans made.
2.Goal oriented: It focuses on defining the goals of an agent (a process), identifying alternative courses of action and deciding the appropriate action plan, which is to be undertaken for reaching the goals.
3.Pervasive: It is pervasive in the sense that it is present in all the segments and is required at all the levels of an agent (a process). Although the scope of planning varies at different levels.
4.Continuous Process: Plans should be updated and are made for a specific term, say for a month, quarter, year and so on. Once that period is over, new plans are drawn, considering the agent's present and future requirements and conditions. Therefore, it is an ongoing process, as the plans are framed, executed and followed by another plan.
5.Intellectual Process: It is a mental exercise at it involves the application of mind, to think, forecast, imagine intelligently and innovate etc.
6.Futuristic: In the planning, we are looking into the future in order to analyze and predict it so that the process can face future challenges effectively.
7.Decision making: Decisions are made regarding the choice of alternative courses of action that can be undertaken to reach the goal. The alternative chosen should be best among all, with the least number of the negative and highest number of positive outcomes.



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
  - Do not work well in other planning domains
- **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**

MESIIA – MIA

10

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

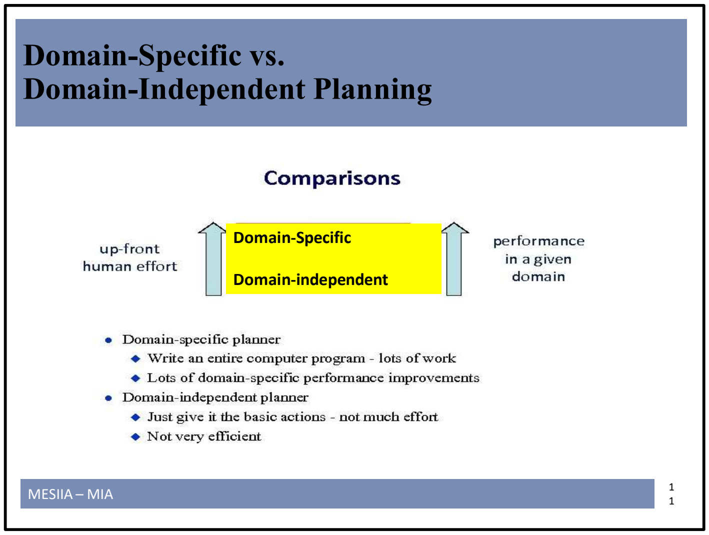
•saves effort; no need to reinvent same techniques for different problems

•leads to general understanding of planning

•contributes to scientific goal of AI

•domain-independent planning complements domain-specific planning

•use domain-independent planning where highly efficient solution is required



Domain-Specific vs. Domain-Independent Planning

Comparisons

up-front human effort

Domain-Specific

Domain-independent

performance in a given domain

- Domain-specific planner
  - Write an entire computer program - lots of work
  - Lots of domain-specific performance improvements
- Domain-independent planner
  - Just give it the basic actions - not much effort
  - Not very efficient

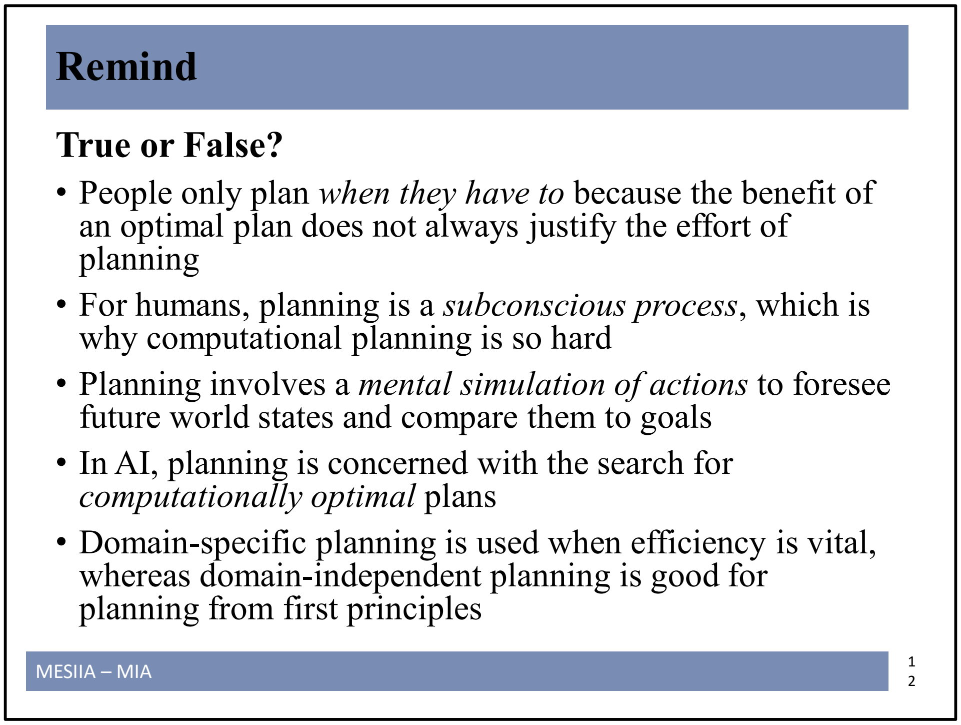
MESIIA – MIA

1
1

Domain-specific needs more efforts, however it improves the performance

- **Domain-specific planner** Write an entire computer program - lots of work Lots of domain-specific performance improvements

- **Domain-independent planner** Just give it the basic actions - not much effort Not very efficient Dana Nau



Remind

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 AI, 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

MESIIA – MIA

12

"People only plan when they have to because the benefit of an optimal plan does not always justify the effort of planning."

True. People often engage in planning when they perceive a clear benefit in terms of achieving their goals or addressing specific challenges. The effort required for planning may vary based on the complexity of the task and the perceived benefits. In some cases, people may choose not to plan if they believe the effort outweighs the potential gains.

"For humans, planning is a subconscious process, which is why computational planning is so hard."

False. While humans engage in both conscious and subconscious cognitive processes related to decision-making and problem-solving, planning can involve conscious, deliberate thought processes. Humans can consciously consider and formulate plans, especially for complex tasks. Computational planning is challenging not because human planning is entirely subconscious but because automating the planning process using algorithms and computers requires addressing complex computational and algorithmic challenges.

"Planning involves a mental simulation of actions to foresee future world states and compare them to goals."

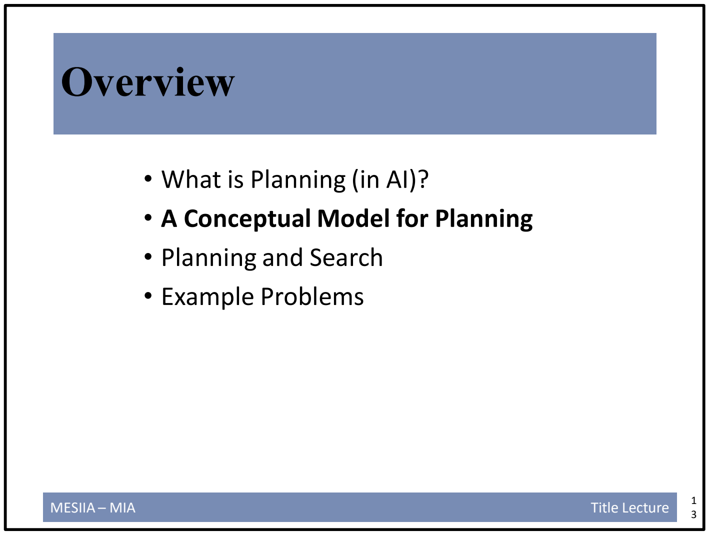
True. Planning often entails mentally simulating potential actions, their consequences, and how they might affect the future state of the world. This mental simulation allows individuals to evaluate different courses of action and assess whether they align with their goals or objectives.

"In AI, planning is concerned with the search for computationally optimal plans."

True. In artificial intelligence, planning involves finding sequences of actions or operations that lead from an initial state to a desired goal state while optimizing for various criteria, such as efficiency, cost, or time. AI planning algorithms aim to identify the most optimal or near-optimal plans within the constraints of the problem.

"Domain-specific planning is used when efficiency is vital, whereas domain-independent planning is good for planning from first principles."

True. Domain-specific planning involves creating plans tailored to a specific problem or domain, often leveraging domain-specific knowledge or heuristics for efficiency. Domain-independent planning, on the other hand, aims to create plans that can be applied across various problem domains without relying on specialized knowledge. Domain-independent planning is useful when planning from first principles or when the problem domain is not well-understood in advance.



Overview

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

Overview

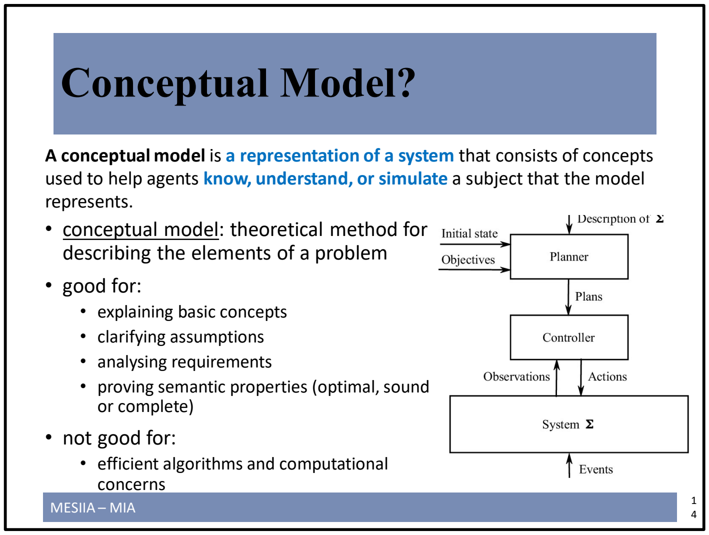
•What is Planning (in AI)?

➢A Conceptual Model for Planning

•now: state-transition systems – formalizing the problem

•Planning and Search

•Example Problems



Conceptual Model?

A conceptual model is **a representation of a system** that consists of concepts used to help agents **know, understand, or simulate** a subject that the model represents.

- conceptual model: theoretical method for describing the elements of a problem
- good for:
  - explaining basic concepts
  - clarifying assumptions
  - analysing requirements
  - proving semantic properties (optimal, sound or complete)
- not good for:
  - efficient algorithms and computational concerns

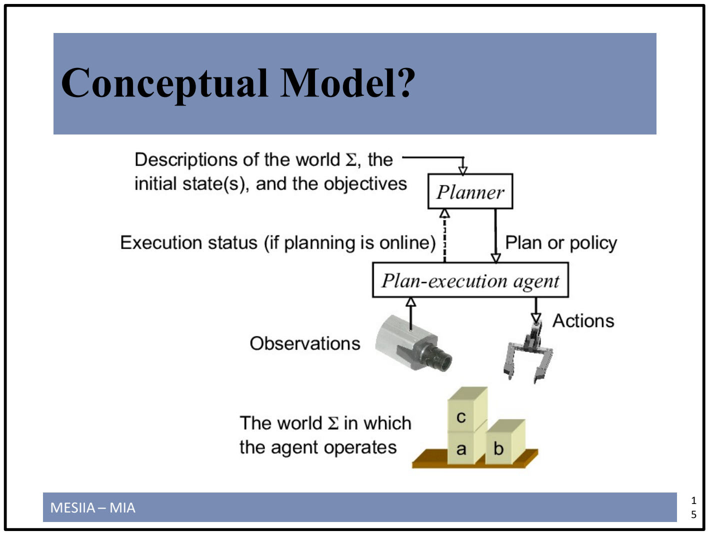
MESIIA – MIA

14

Why a Conceptual Model?

- **conceptual model: theoretical method for describing the elements of a problem**
- **good for:**
  - **explaining basic concepts**: what are the objects to be manipulated during problem-solving?
  - **clarifying assumptions**: what are the constraints imposed by this model?
  - **analysing requirements**: what representations do we need for the objects?
  - **proving semantic properties**: when is an algorithm sound or complete?
    - Three main Properties of Planning Algorithms are that planner should be sound, complete and optimal:
      - Optimality, this expression can be complicated. For instance, we may have many factors affecting optimality of a subway trip (stations with escalators, crowds on trains, etc.). To find the "Best", something is very complex.
      - An algorithm is sound if, anytime it returns an answer, that answer is true." means the same as "Basically, soundness (of an algorithm) means that the algorithm doesn't yield any results that are untrue."
      - An algorithm is complete if it guarantees to return a correct answer for any arbitrary input (or, if no answer exists, it guarantees to return failure).
      - An algorithm can be said to be optimal if the function that describes its time complexity in the worst case is a lower bound of the function that describes the time complexity in the worst case of a problem that the algorithm in question solves. " asymptotically solve the problem in less time"

- **not good for:**
  - **efficient algorithms and computational concerns**



Conceptual Model?

Descriptions of the world Σ, the initial state(s), and the objectives

Planner

Execution status (if planning is online)

Plan or policy

Plan-execution agent

Observations

Actions

The world Σ in which the agent operates

c

a

b

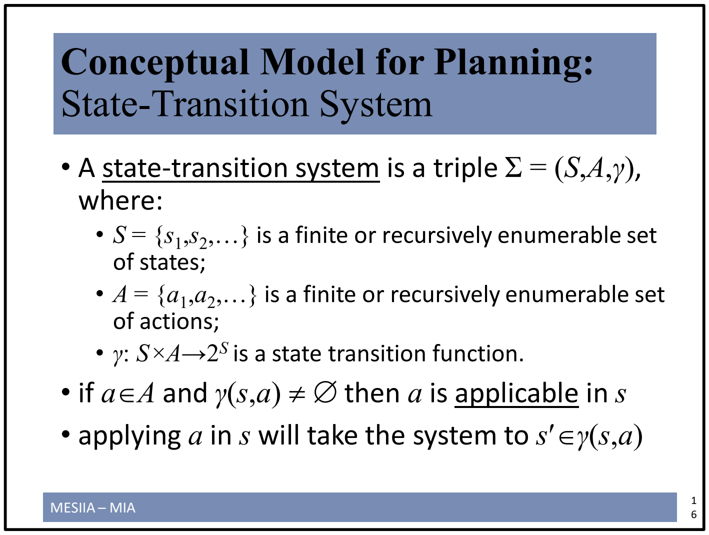
MESIIA – MIA

15

Conceptual model of AI planning

The planner’s input includes descriptions of *Σ*, the *initial* state(s) that *Σ* might be in before the plan-execution agent performs any actions, and the desired objectives (e.g., to reach a set of states that satisfies a given *goal condition*, or to perform a specified task, or a set of states that the world should be kept in or kept out of, or a partially ordered set of states that we might want the world to go through). If the planning is being done online (i.e., if planning and plan execution are going on at the same time), the planner’s input will also include feedback about the current execution status of the plan or policy.

The planner’s output consists of either a *plan* (a linear sequence of actions for the agent to perform) or a *policy* (a set of state-action pairs with at most one action for each state).



Conceptual Model for Planning: State-Transition System

- A state-transition system is a triple $\Sigma = (S,A,\gamma)$, 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;
  - γ: $S\times A\rightarrow 2^S$ is a state transition function.
- if $a\in A$ and $\gamma(s,a) \neq \varnothing$ then a is applicable in s
- applying a in s will take the system to $s'\in\gamma(s,a)$

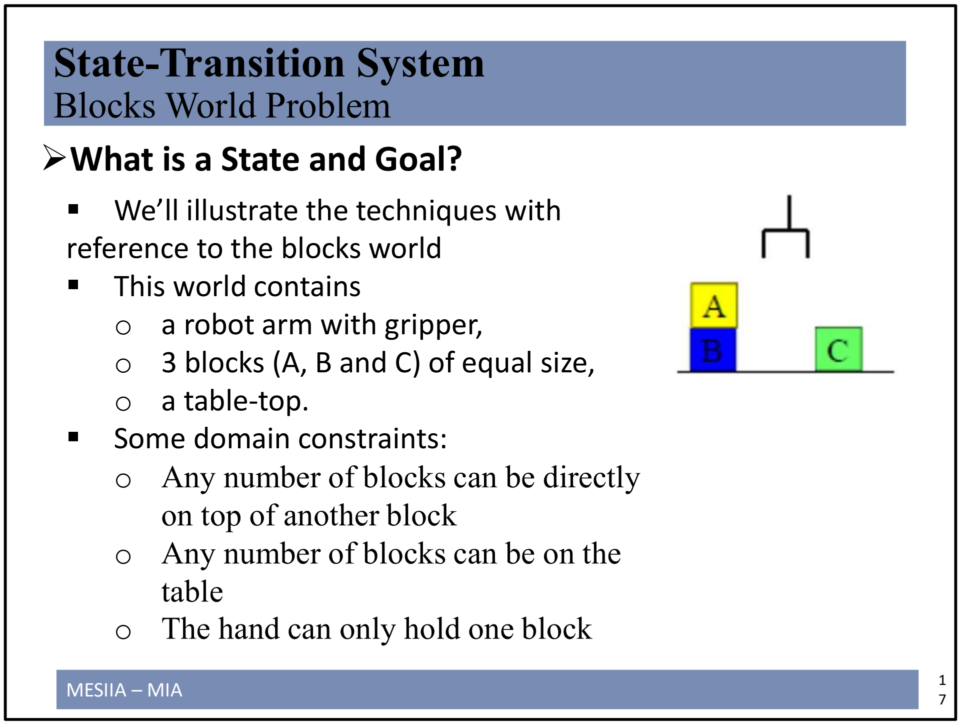
Conceptual Model for Planning: State-Transition Systems

•A state-transition system is sometimes a 4-tuple Σ=(S,A,E,γ), where:

- •a general model for a dynamic system, common to other areas of computer science; aka. dynamic-event system
- **•$S = \{s_1,s_2,\ldots\}$ is a finite or recursively enumerable set of states;**
  - •the possible states the world can be in
- **•$A = \{a_1,a_2,\ldots\}$ is a finite or recursively enumerable set of actions;**
  - •the actions that can be performed by some agent in the world, transitions are controlled by the plan executor
- **•$E = \{e_1,e_2,\ldots\}$ is a finite or recursively enumerable set of events; and**
  - •the events that can occur in the world, transitions that are contingent (correspond to the internal dynamics of the system)
- **•γ: $S\times(A\cup E)\rightarrow 2^S$ is a state transition function.**
  - •notation: 2^S=powerset of S; maps to a set of states
  - •the function describing how the world evolves when actions or events occur
- •note: model does not allow for parallelism between actions and/or events

•if $a\in A$ and $\gamma(s,a) \neq \varnothing$ then a is applicable in s

•applying a in s will take the system to $s'\in\gamma(s,a)$



State-Transition System
Blocks World Problem

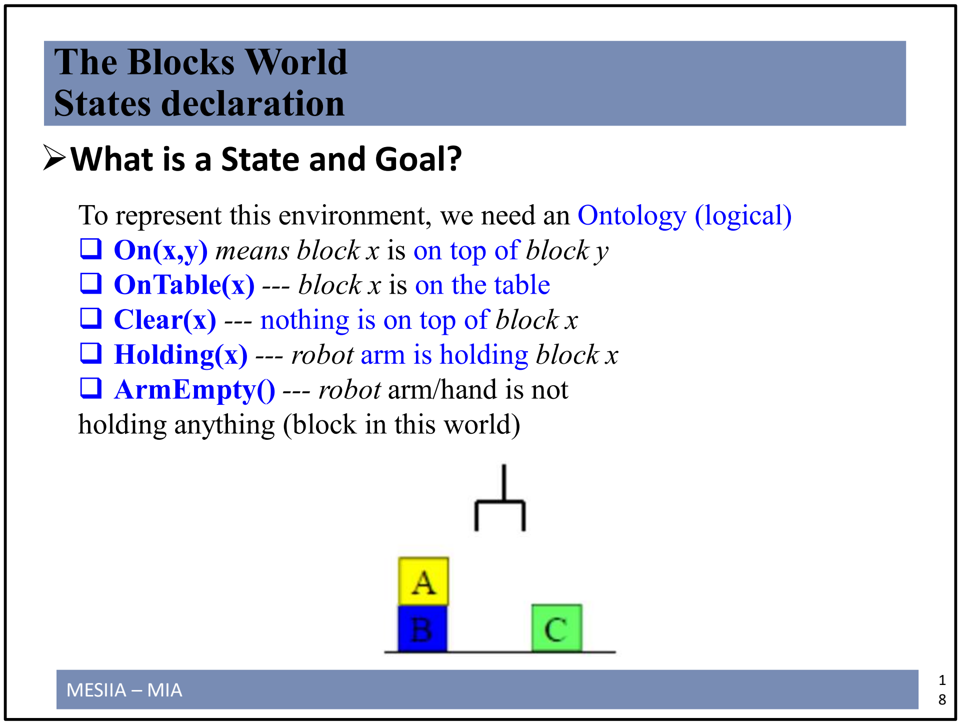
➢**What is a State and Goal?**

- We'll illustrate the techniques with reference to the blocks world
- This world contains
  - a robot arm with gripper,
  - 3 blocks (A, B and C) of equal size,
  - a table-top.
- Some domain constraints:
  - Any number of blocks can be directly on top of another block
  - Any number of blocks can be on the table
  - The hand can only hold one block

MESIIA – MIA 17

Here, the world belief definition of the blocks world problem,
1) a robot arm with gripper,
2) 2) 3 blocks (A, B and C) of equal size,
3) 3) a table-top.

The domain constraints should be taken into account are:
1) Only one block can be directly on top of another block,
2) 2) Any number of blocks can be on the table, and
3) The hand can only hold one block



The Blocks World
States declaration

➢**What is a State and Goal?**

To represent this environment, we need an Ontology (logical)

❑ **On(x,y)** *means block x* is on top of *block y*

❑ **OnTable(x)** --- *block x* is on the table

❑ **Clear(x)** --- nothing is on top of *block x*

❑ **Holding(x)** --- *robot* arm is holding *block x*

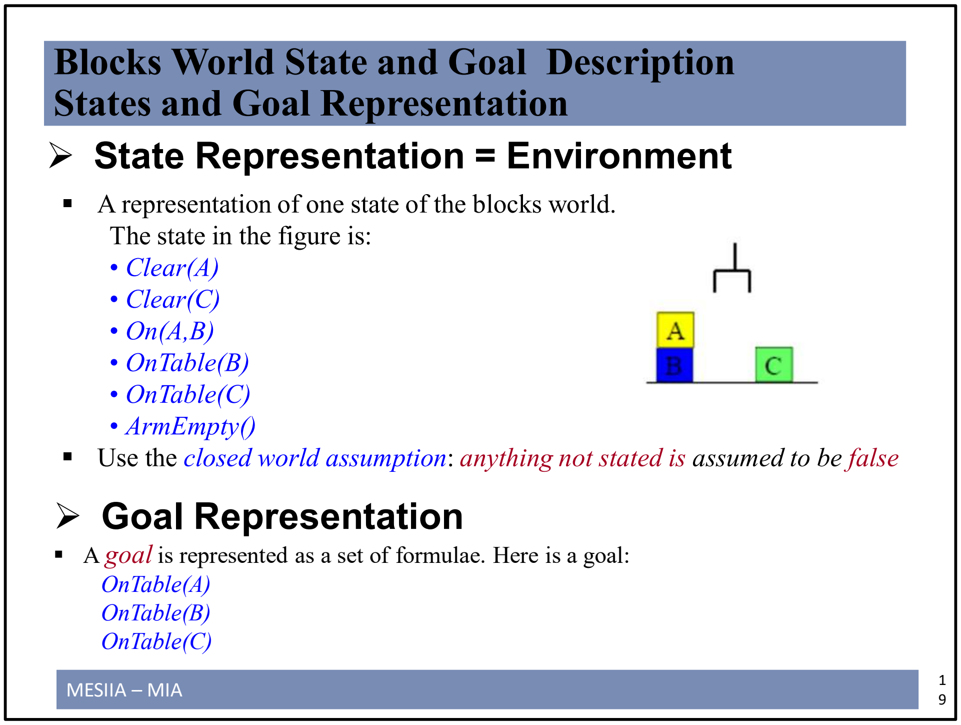
❑ **ArmEmpty()** --- *robot* arm/hand is not holding anything (block in this world)

MESIIA – MIA

18

In philosophy, **ontology is** the study of what exists in the environment.
In **AI**, an **ontology is** a specification of the meanings of the symbols in an information system.
It **is** a specification of what individuals and relationships **are** assumed to exist and what terminology **is** used for them.

For instance,
On function accepts two parameters to check if the block x over the block y.
OnTable function to check if x block on the table.
Clear function is to be sure nothing over the block x.
Holding function is to be sure the robot arm holding a block x, in turn
ArmEmpty to be sure nothing holding by the arm.



Blocks World State and Goal Description
States and Goal Representation

➢ State Representation = Environment

- A representation of one state of the blocks world.

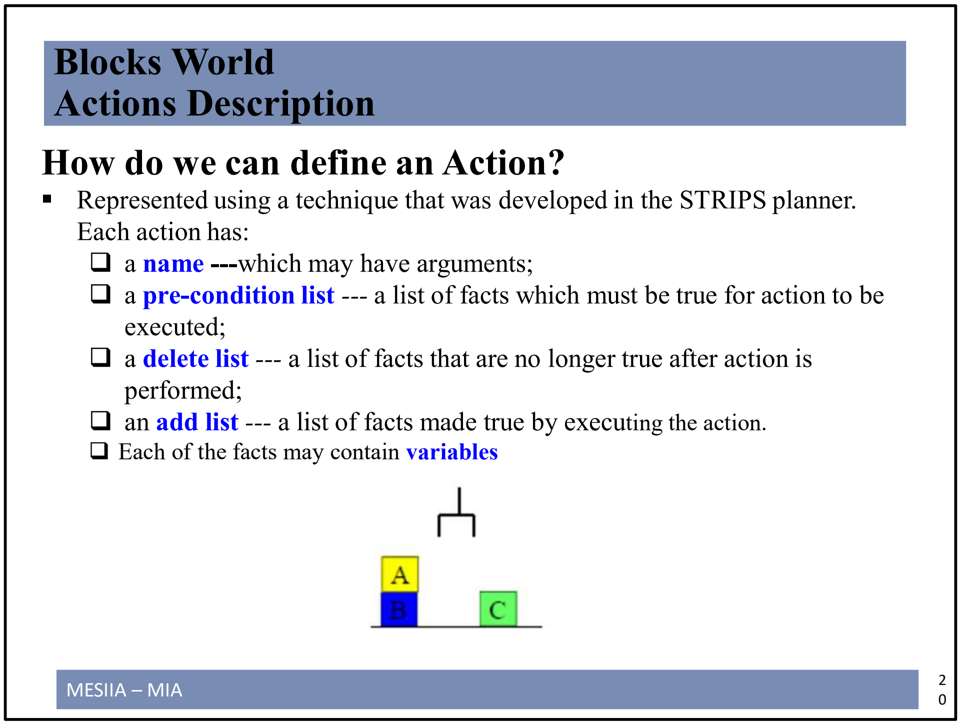
  The state in the figure is:
  - *Clear(A)*
  - *Clear(C)*
  - *On(A,B)*
  - *OnTable(B)*
  - *OnTable(C)*
  - *ArmEmpty()*

- Use the *closed world assumption*: *anything not stated is assumed to be false*

➢ Goal Representation

- A *goal* is represented as a set of formulae. Here is a goal:

  OnTable(A)
  OnTable(B)
  OnTable(C)



Blocks World
Actions Description

How do we can define an Action?

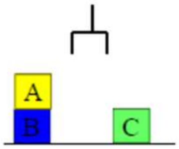
- Represented using a technique that was developed in the STRIPS planner. Each action has:
  - ❑ a **name** **---**which may have arguments;
  - ❑ a **pre-condition list** --- a list of facts which must be true for action to be executed;
  - ❑ a **delete list** --- a list of facts that are no longer true after action is performed;
  - ❑ an **add list** --- a list of facts made true by executing the action.
  - ❑ Each of the facts may contain **variables**

The action can defined by 1) its name, 2) pre-condition list (an example of pre-conditions list is: assume you want to put block C on a block B. So you need to be sure that the top of the block B is empty, *clear (B)), 3) delete list and 4) add list. So what are the main actions we need them to define the block world??*

Blocks World Actions Description

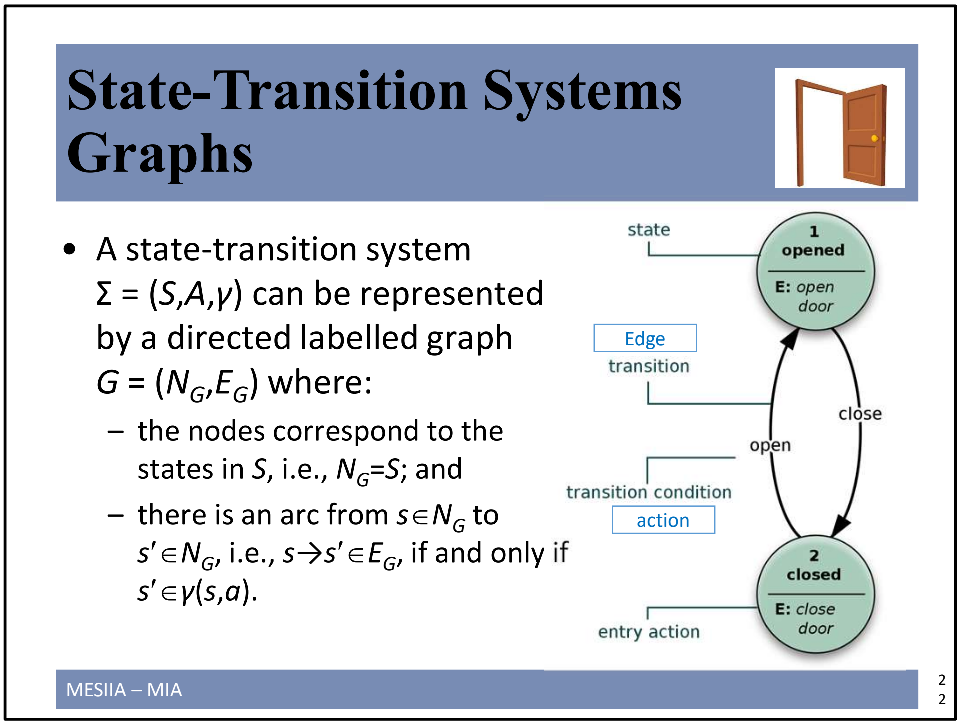
Action/Operator Representation

- Basic operations
  - stack(X,Y): put block X on block Y
  - unstack(X,Y): remove block X from block Y
  - pickup(X): pickup block X from the table
  - putdown(X): put block X on the table



- Each operator is represented by facts that describe the state of the world before and changes to the world after an action is performed.
  - a list of **preconditions**
  - a list of new **facts to be added** (add-effects)
  - a list of **facts to be deleted** (delete-effects)
  - optionally, a set of (simple) variable **constraints**

We will study this example in the Lab



State-Transition Systems Graphs

- A state-transition system $\Sigma = (S,A,\gamma)$ 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., $s \rightarrow s' \in E_G$, if and only if $s' \in \gamma(s,a)$.

state
1
opened
E: open door
Edge
transition
close
open
transition condition
action
2
closed
E: close door
entry action

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:

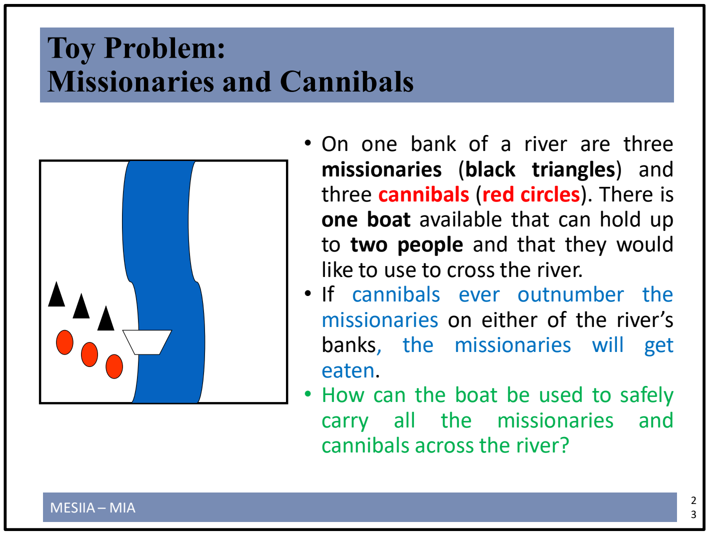
•the nodes correspond to the states in S, i.e., $N_G=S$; and

•nodes correspond to world states

•there is an arc from $s \in N_G$ to $s' \in N_G$, i.e., $s \rightarrow s' \in E_G$, with label $a \in (A \cup E)$ if and only if $s' \in \gamma(s,a)$.

•there is an arc if there is an action or event that transforms one state into the other (called a state transition)

•the label of that arc is that action or event



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 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?

Toy Problem: Missionaries and Cannibals

•On one bank of a river are three missionaries (black triangles) and three cannibals (red circles).

  •description of initial state and graphical representation; other states must follow the same pattern

•There is one boat available that can hold up to two people and that they would like to use to cross the river.

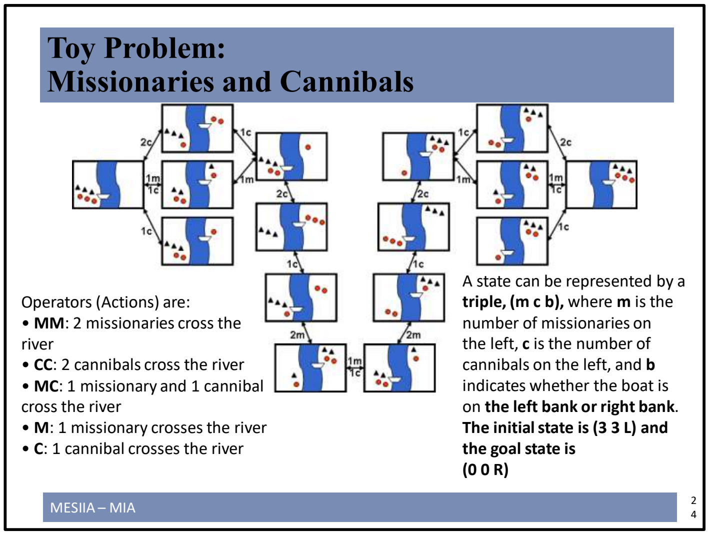
  •more state description: boat; implicitly: description of possible action: cross river using boat

•If the cannibals ever outnumber the missionaries on either of the river's banks, the missionaries will get eaten.

  •constraint on all states; must not be violated

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

  •problem: find a sequence of action that brings about another state



Toy Problem: Missionaries and Cannibals

Operators (Actions) are:
- **MM**: 2 missionaries cross the river
- **CC**: 2 cannibals cross the river
- **MC**: 1 missionary and 1 cannibal cross the river
- **M**: 1 missionary crosses the river
- **C**: 1 cannibal crosses the river

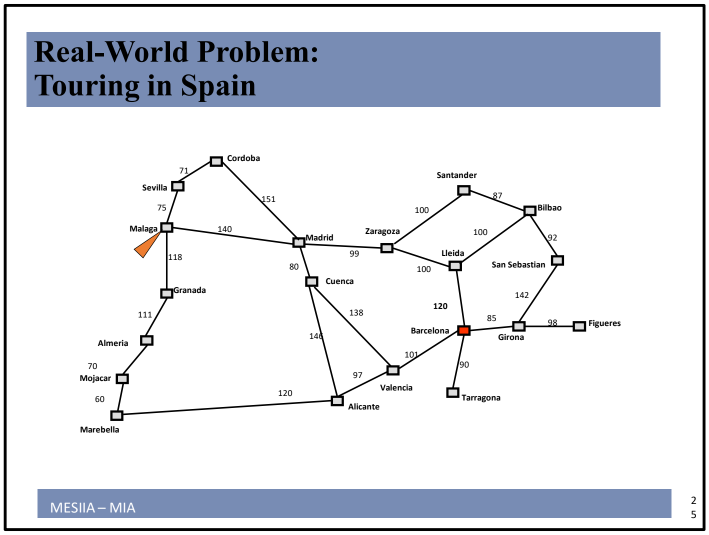
A state can be represented by a **triple, (m c b),** where **m** is the number of missionaries on the left, **c** is the number of cannibals on the left, and **b** indicates whether the boat is on **the left bank or right bank**. **The initial state is (3 3 L) and the goal state is (0 0 R)**

MESIIA – MIA

24

State-Transition Graph Example: Missionaries and Cannibals

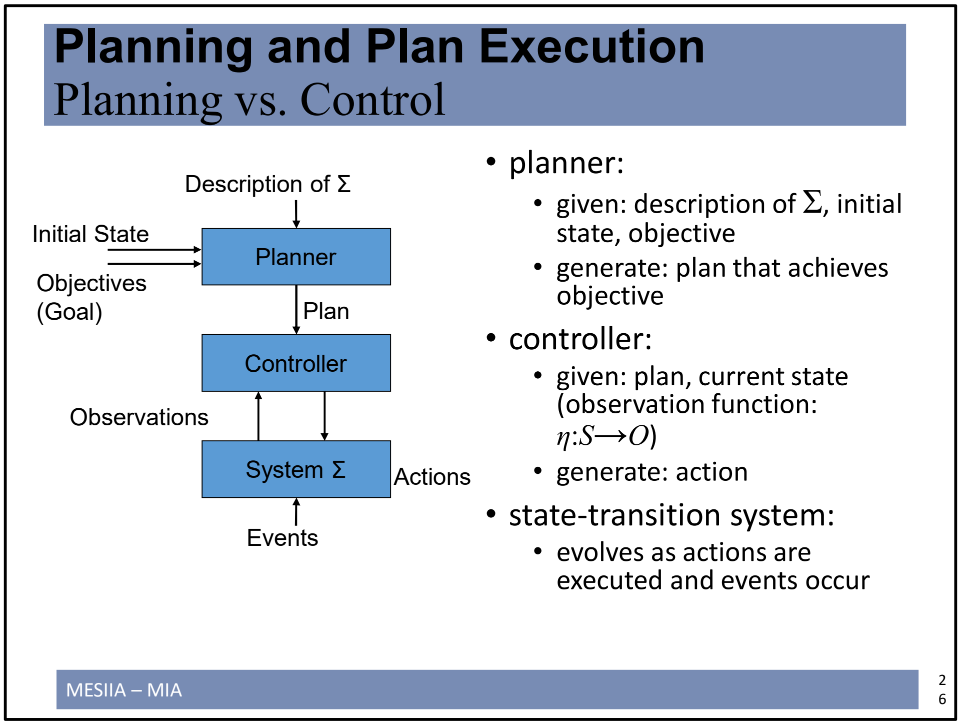
- Define the initial state: 3 C, 3 M and a boat on the right hand side of the river
- Define the goal state: 3C, 3M and a boat on the right hand side of the river
- Available actions are: 2c -> move 2 cannibals, 2m -> move 2 missionaries, 1c1m -> move a missionary and a cannibal, 1c -> move a cannibal, and 1m -> move a missionary.
- Events is empty.



Real-World Problem: Touring in Spain

Real-World Problem: Touring in Spain

- shown: rough map of Spain
- initial state: on vacation in Malaga, Spain
- goal? actions? -- “Touring Spain” cannot readily be described in terms of possible actions, goals, and path cost



Planning and Plan Execution
Planning vs. Control

- 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

Planning and Plan Execution

•**planner:**

•**given: description of Σ, initial state, objective**

•**generate: plan that achieves objective**

•planner works offline: relies on formal description of Σ

•**controller:**

•**given: plan, current state (observation function: $\eta:S\rightarrow O$)**

•partial knowledge of controller about world modelled through observation function

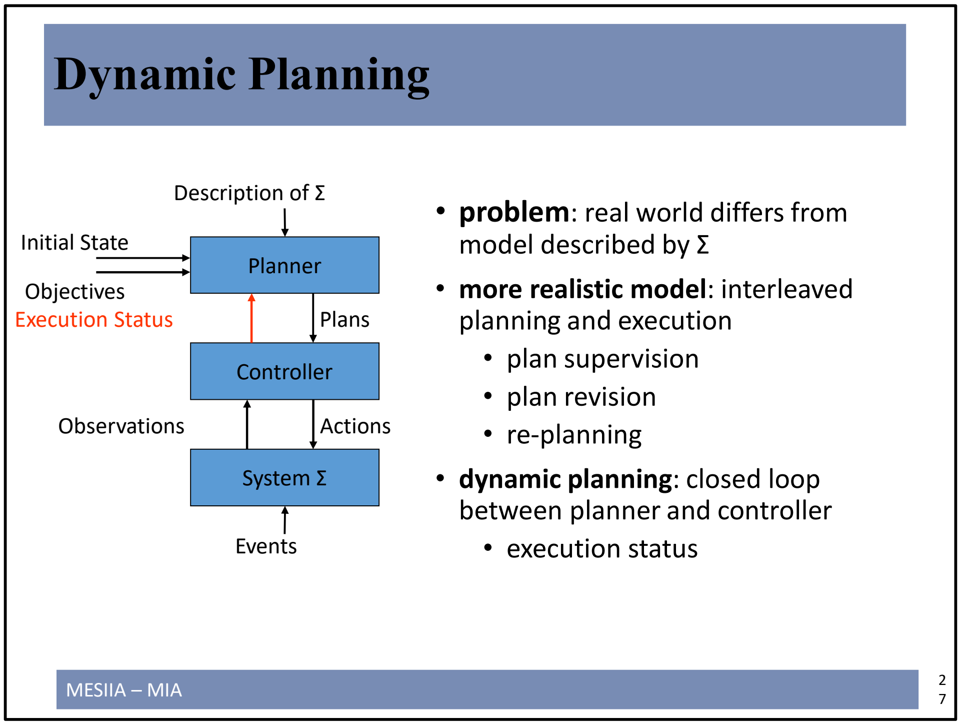
•O = set of possible observations (e.g., subset); input for controller

•**generate: action**

•controller works online: along with the dynamics of Σ

•**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

Dynamic Planning

•problem: physical system differs from model described by Σ

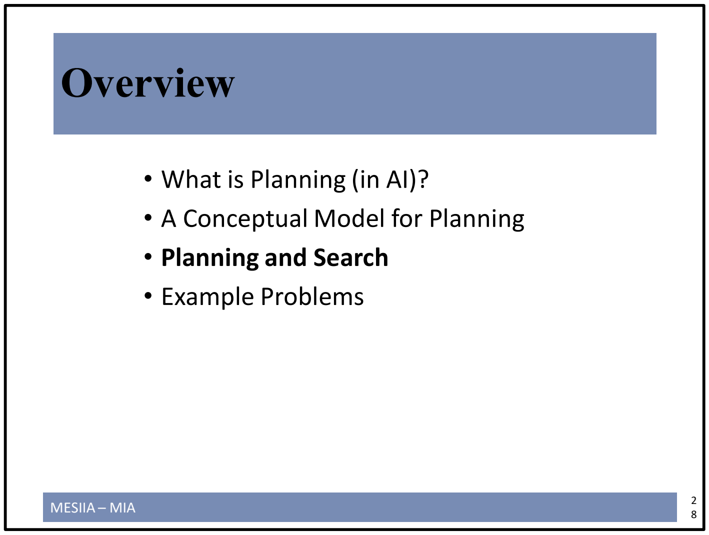
- •planner only has access to model (description of Σ)
- •controller must cope with differences between Σ and real world

•more realistic model: interleaved planning and execution

- •**plan supervision**: detect when observations differ from expected results
- •**plan revision**: adapt existing plan to new circumstances
- •**re-planning**: generate a new plan from current (initial) state

•dynamic planning: closed loop between planner and controller

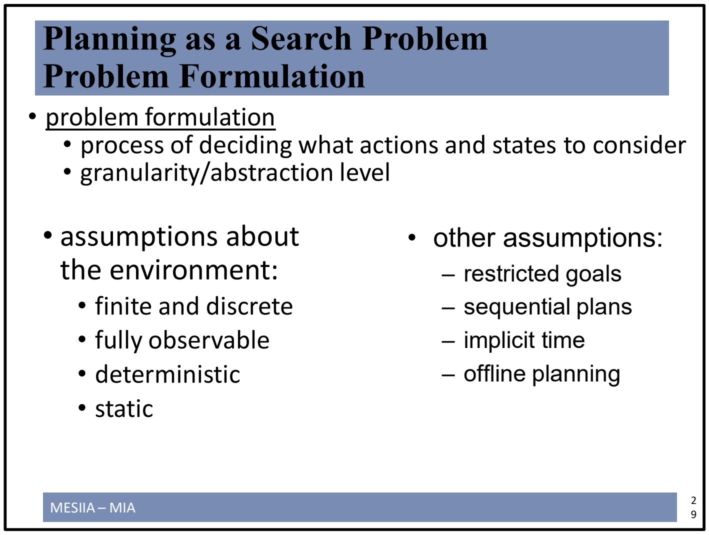
- •**execution status**



Overview

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

Planning is a search process to select the best (optimal) set of available actions.



Planning as a Search Problem
Problem Formulation

- <u>problem formulation</u>
  - 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

MESIIA – MIA

29

Problem Formulation

•**<u>problem formulation</u>**

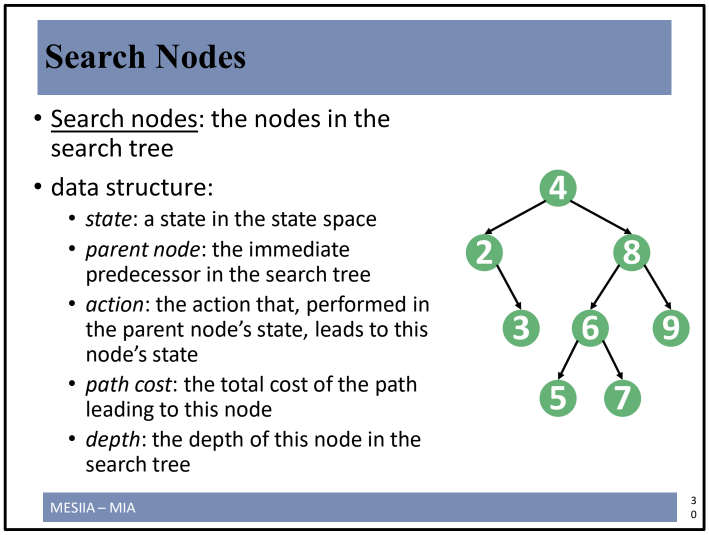
- •**process of deciding what actions and states to consider**
- •**granularity/abstraction level**
- •Touring Spain example:
  - •possible actions: turn steering wheel by one degree left/right; move left foot … → too much (irrelevant) detail, uncertainty, too many steps in solution
  - •possible action: drive to Barcelona → How to execute such an action?
  - •best level of granularity: drive to another major town to which there is a direct road

•**assumptions about the environment:**

- •**finite and discrete**: we can enumerate all the possible actions and states (else: continuous)
- •**fully observable**: agent can see what current state it is in
- •**deterministic**: applying an action in a given state leads to exactly one (usually different) state
- •**static**: environment does not change while we think (no events and no actions)

•**other assumptions:**

- •**restricted goals:** given as an explicit goal state s_g or set of goal states S_g
- •**sequential plans:** solution plan is a linearly ordered finite sequence of actions
- •**implicit time:** actions and events have no duration
- •**offline planning:** planner is not concerned with changes of Σ while 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

Search Nodes

•<u>**search nodes**</u>**: the nodes in the search tree**

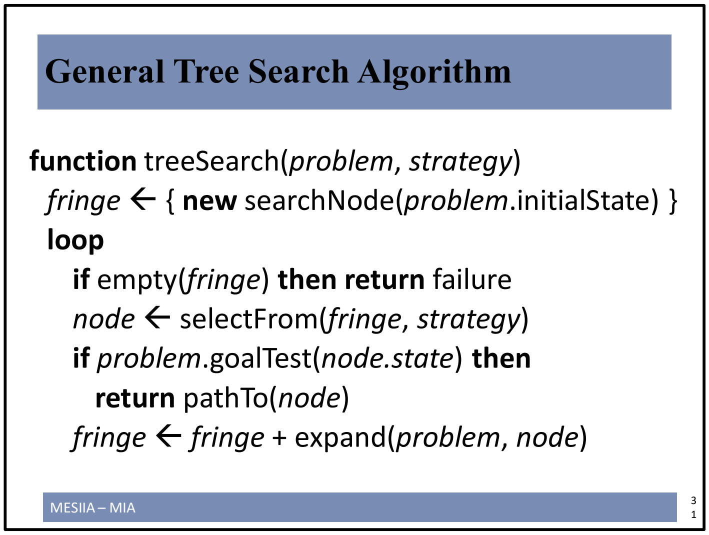
- •node is a bookkeeping structure in a search tree

•**data structure:**

- •***state*: a state in the state space**
  - •state (vs. node) corresponds to a configuration of the world
  - •two nodes may contain equal states
- •***parent node*: the immediate predecessor in the search tree**
  - •nodes are on paths (defined by parent nodes)
- •***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**

•alternative: representing paths only (sequences of actions):

- •possible, but state provides direct access to valuable information that might be expensive to regenerate all the time



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)
```

MESIIA – MIA

31

General Tree Search Algorithm

function treeSearch(*problem*, *strategy*)

•find a solution to the given problem while expanding nodes according to the given strategy

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

•*fringe*: set of known states; initially just initial state

loop

•possibly infinite loop expands nodes

if empty(*fringe*) then return failure

•complete tree explored; no goal state found

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

•select node from fringe according to search control strategy; the node will not be selected again

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

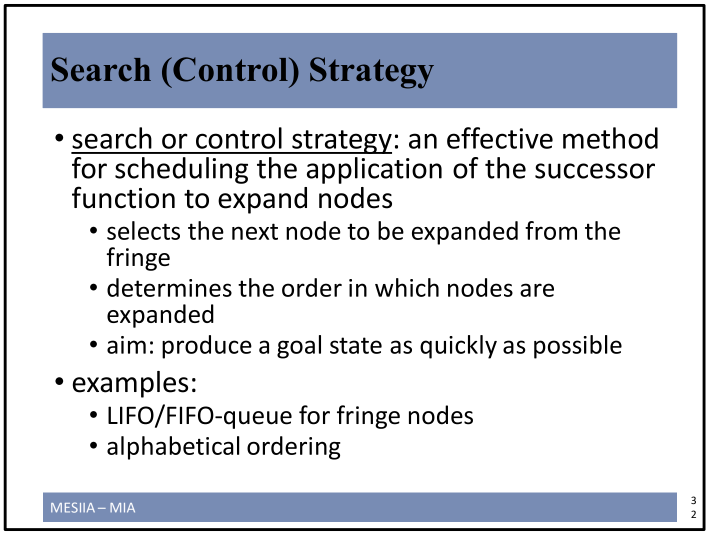
•goal test before expansion: to avoid trick problem like “get from Malaga to Malaga”

return pathTo(*node*)

•success: goal node found

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

•otherwise: add new nodes to the fringe and continue loop



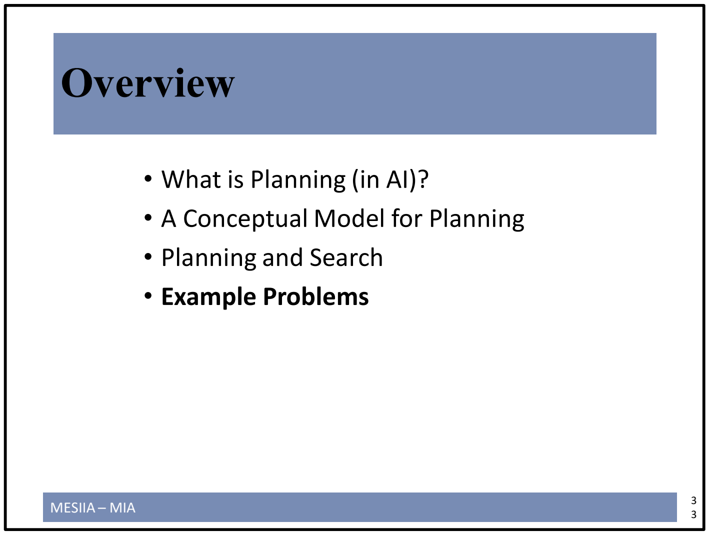
Search (Control) Strategy

- 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

Search (Control) Strategy

- **search or control strategy: an effective method for scheduling the application of the successor function to expand nodes**
  - removes non-determinism from search method
  - **selects the next node to be expanded from the fringe**
    - closed nodes never need to be expanded again
  - **determines the order in which nodes are expanded**
    - exact order makes method deterministic
  - **aim: produce a goal state as quickly as possible**
    - strategy that produces goal state quicker is usually considered better
- **examples:**
  - **LIFO/FIFO-queue for fringe nodes** (two fundamental search strategies)
  - **alphabetical ordering**

- remark: complete search tree is usually too large to fit into memory, strategy determines which part to generate

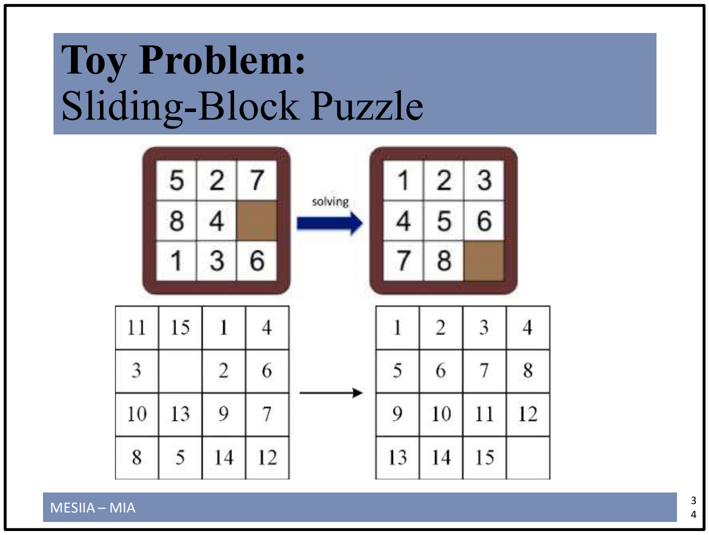


Overview

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

Overview

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



Toy Problem:
Sliding-Block Puzzle

5	2	7
8	4	▨
1	3	6

1	2	3
4	5	6
7	8	▨

11	15	1	4
3		2	6
10	13	9	7
8	5	14	12

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

Toy Problem: Sliding-Block Puzzle

•[images]

•The 8-Puzzle

•problem description:

•states: location of each tile and the blank (9!/2 = 181440 states)

•initial state: any (reachable) state, e.g. one shown on left (worst case: solution requires at least 31 steps)

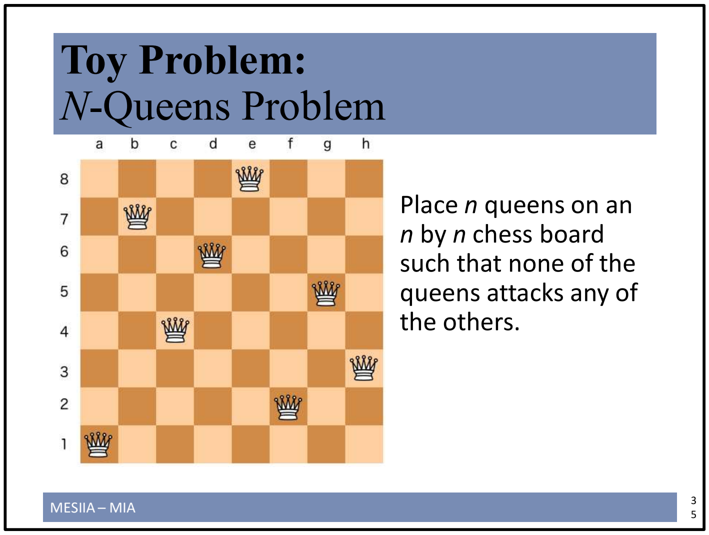
•actions: good formulation: move blank left/right/up/down

•goal test: goal state (shown right)

•step cost: 1

•generalization: sliding-tile puzzles, e.g. 15-puzzle, 24-puzzle

•problem class is NP-complete



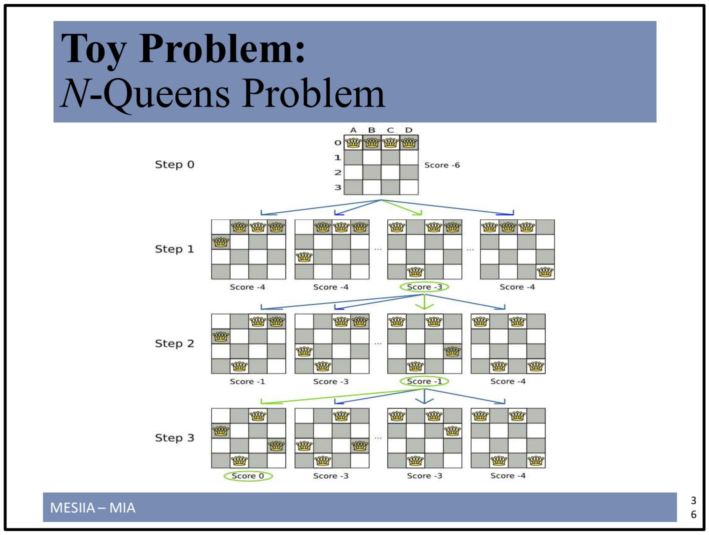
Toy Problem:
N-Queens Problem

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

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.

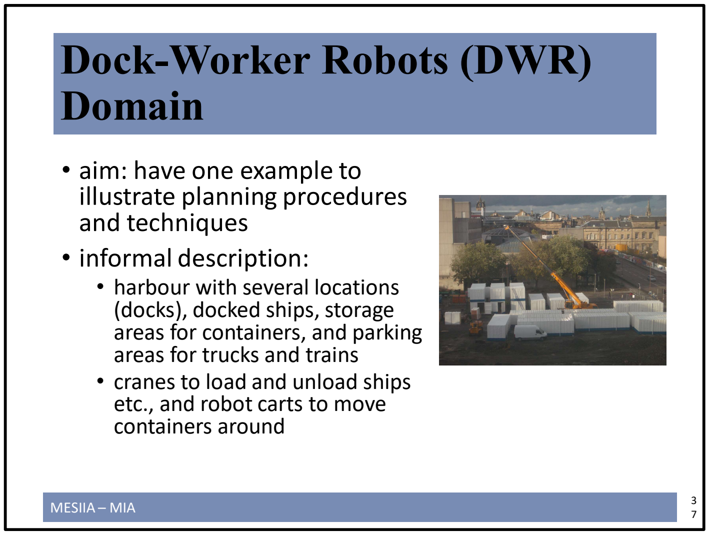
- •problem: no row, column, or diagonal must contain more than one queen
- •path cost irrelevant; looking for goal state
- •configuration shown is a goal state or not?



Toy Problem: *N*-Queens Problem

Toy Problem: *N*-Queens Problem

- Place *4* queens on an *4* by *4* chess board such that none of the queens attacks any of the others.
- Initial state step 0 and goal is step 3



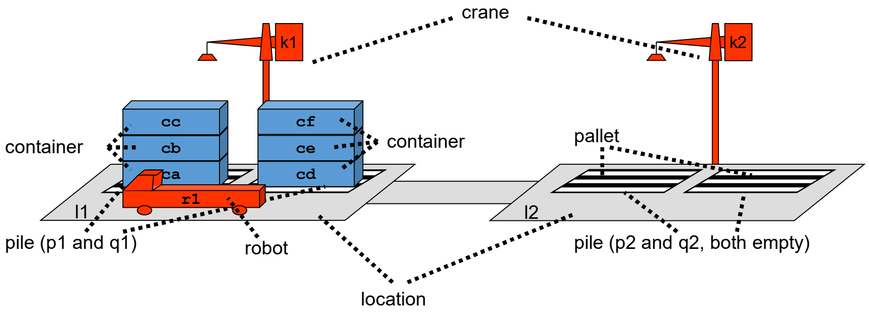
Dock-Worker Robots (DWR) Domain

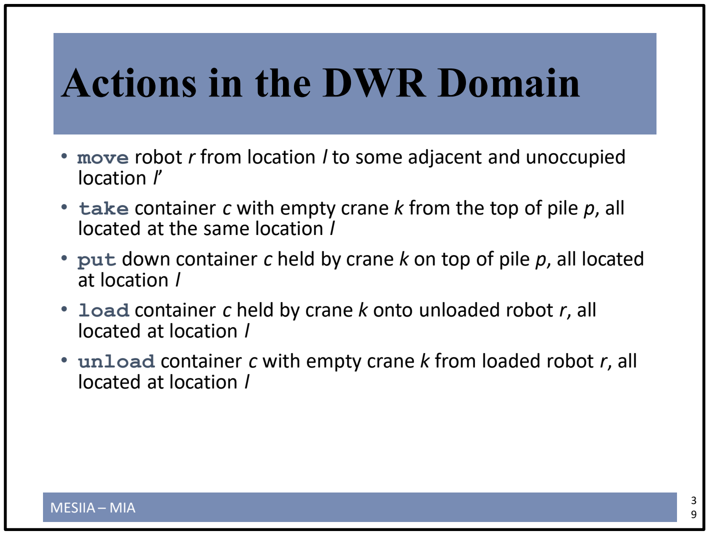
- aim: have one example to illustrate planning procedures and techniques
- informal description:
  - harbour with several locations (docks), docked ships, storage areas for containers, and parking areas for trucks and trains
  - cranes to load and unload ships etc., and robot carts to move containers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

The Dock-Worker Robots (DWR) Domain

- **aim: have one example to illustrate planning procedures and techniques**
  - problem must be nontrivial to be interesting, but not too much overhead to introduce problem
- **informal description:**
  - generalization of earlier example (state transition graph)
  - **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**
- simplified and enriched version of this domain will be introduced later
- approach: use first-order predicate logic as representation

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*

MESIIA – MIA

39

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*

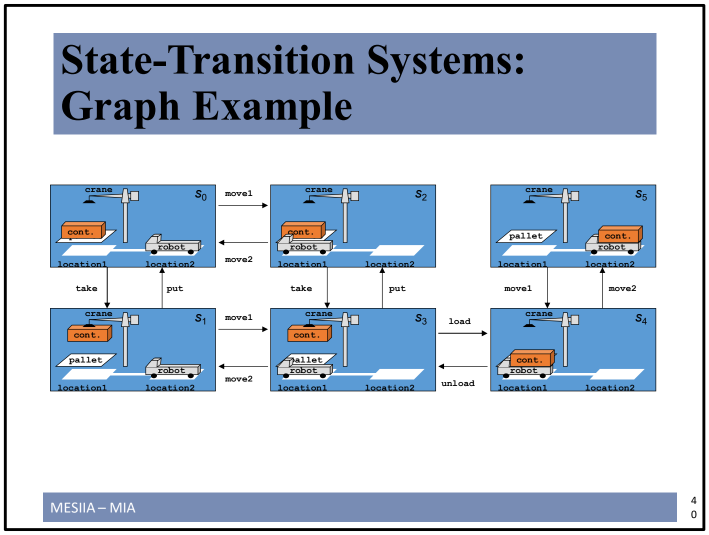
•put down container *c* held by crane *k* on top of pile *p*, all located at location *l*

•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*

•formal specifications will follow when we have introduced a formal action description language

•problem: how to represent actions formally? first-order logic?

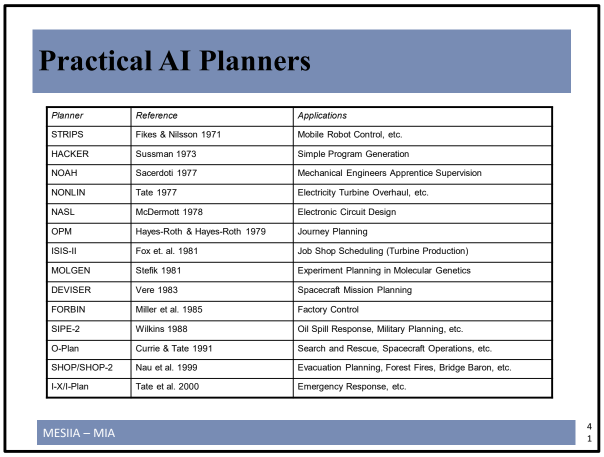


State-Transition Systems: Graph Example

MESIIA – MIA

State-Transition Systems: Graph Example

- states: s_0 to s_5
  - objects: robot, crane, container, pallet, two locations
- actions:
  - crane can take/put the container from the pallet/onto the pallet
  - crane can load/unload the container from the robot
  - robot can drive to either location (with or without the container loaded)
- no events
- state transition function: arcs shown in graph
- note: state transitions are deterministic: each action leads to at most one other state



Practical AI Planners

Planner	*Reference*	*Applications*
STRIPS	Fikes & Nilsson 1971	Mobile Robot Control, etc.
HACKER	Sussman 1973	Simple Program Generation
NOAH	Sacerdoti 1977	Mechanical Engineers Apprentice Supervision
NONLIN	Tate 1977	Electricity Turbine Overhaul, etc.
NASL	McDermott 1978	Electronic Circuit Design
OPM	Hayes-Roth & Hayes-Roth 1979	Journey Planning
ISIS-II	Fox et. al. 1981	Job Shop Scheduling (Turbine Production)
MOLGEN	Stefik 1981	Experiment Planning in Molecular Genetics
DEVISER	Vere 1983	Spacecraft Mission Planning
FORBIN	Miller et al. 1985	Factory Control
SIPE-2	Wilkins 1988	Oil Spill Response, Military Planning, etc.
O-Plan	Currie & Tate 1991	Search and Rescue, Spacecraft Operations, etc.
SHOP/SHOP-2	Nau et al. 1999	Evacuation Planning, Forest Fires, Bridge Baron, etc.
I-X/I-Plan	Tate et al. 2000	Emergency Response, etc.

SIPE-2 – System for Interactive Planning and Execution

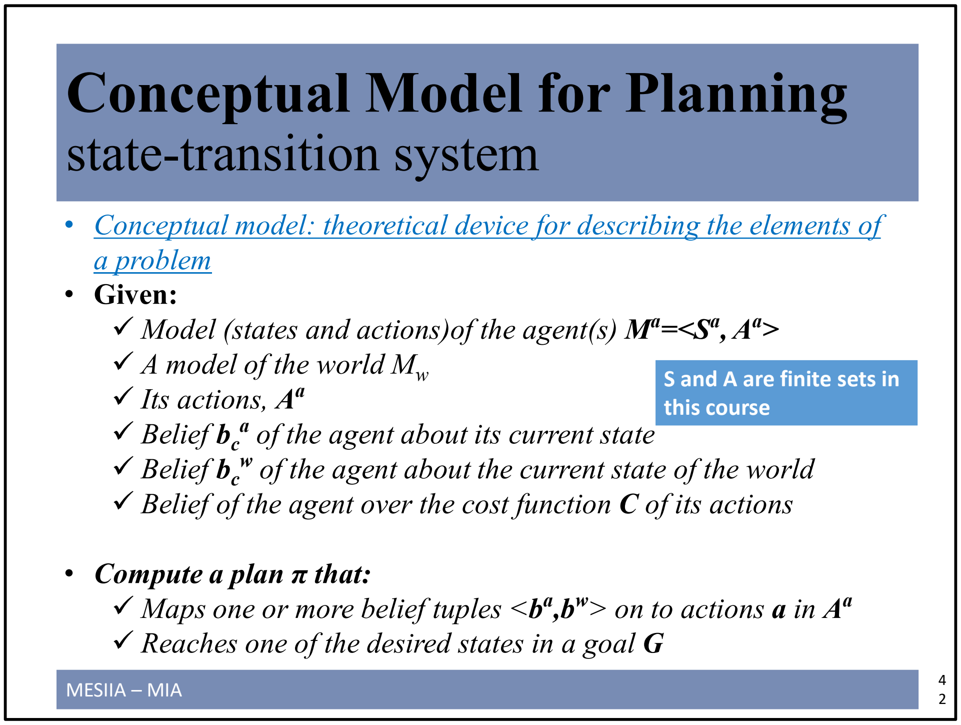
MPA – Multiagent Planning Architecture – used as the basis for SIPE-2. Multiple planning agents which communicate plans and refine parts relevant to their area of expertise or responsibility.

O-Plan – Open Planning Architecture (multiple agent planning framework) – sharing of partial plans and plan fragments between different types of agents involved in b the panning process.

Wilkins, D.E. (1988) Practical Planning: Extending the Classical AI Planning Paradigm, Morgan Kaufmann Publishers Inc., San Francisco, CA, 1988.

Wilkins D.E. and desJardins, M. (1993) Temporal Reasoning in the SIPE-2 Planner, AAAI Spring Symposium, 1993.
http://www.ai.sri.com/~wilkins/bib.html

Dana Nau, Applications of SHOP and SHOP-2
Nau, D., Au, T-C., Ilghami, O., Kuter, U., Wu, D.,Yaman, F., Muñoz-Avila, H., and Murdock, J.W. (2005) Applications of SHOP and SHOP2, IEEE Intelligent Systems, March-April 2005, Vol. 20, No. 2, pp.34-41, Computer Society.
http://www.cs.utexas.edu/~chiu/papers/Nau05shop2.pdf



Conceptual Model for Planning

state-transition system

- *Conceptual model: theoretical device for describing the elements of a problem*
- **Given:**
  - ✓ *Model (states and actions)of the agent(s)* $\boldsymbol{M^a}$=<$\boldsymbol{S^a}$, $\boldsymbol{A^a}$>
  - ✓ *A model of the world* M_w
  - ✓ *Its actions,* $\boldsymbol{A^a}$
  - ✓ *Belief* $\boldsymbol{b_c^a}$ *of the agent about its current state*
  - ✓ *Belief* $\boldsymbol{b_c^w}$ *of the agent about the current state of the world*
  - ✓ *Belief of the agent over the cost function **C** of its actions*

S and A are finite sets in this course

- ***Compute a plan π that:***
  - ✓ *Maps one or more belief tuples <*$\boldsymbol{b^a}$,$\boldsymbol{b^w}$*> on to actions **a** in* $\boldsymbol{A^a}$
  - ✓ *Reaches one of the desired states in a goal **G***

MESIIA – MIA

4
2

Why a Conceptual Model?

•<u>conceptual model</u>: theoretical device for describing the elements of a problem

•good for:

•explaining basic concepts: what are the objects to be manipulated during problem-solving?

•clarifying assumptions: what are the constraints imposed by this model?

•analysing requirements: what representations do we need for the objects?

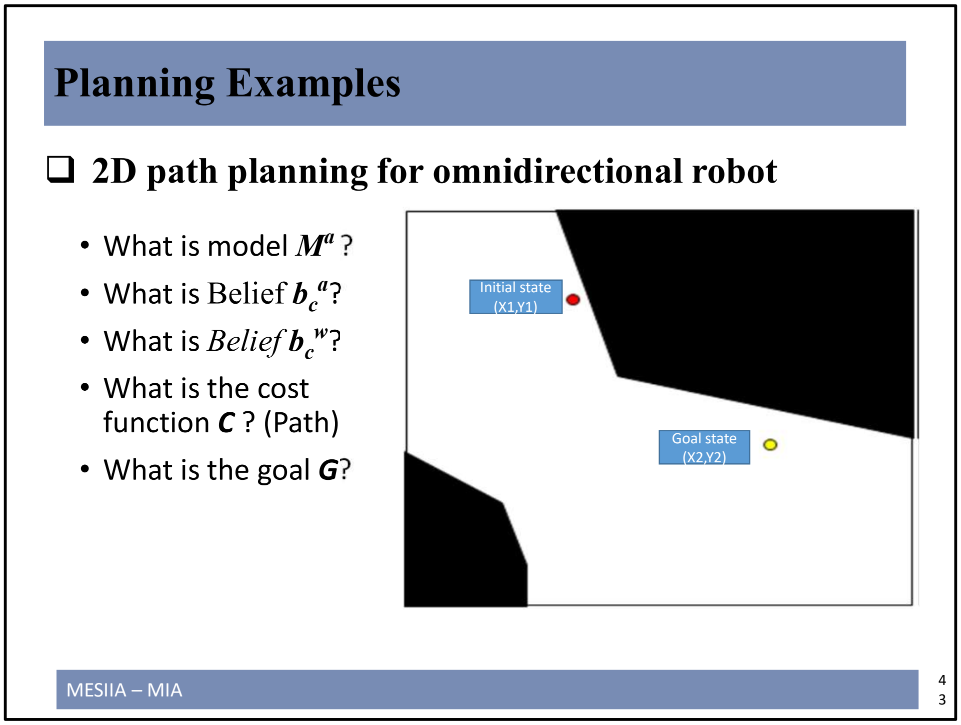
•proving semantic properties: when is an algorithm sound or complete?

•not good for:

•efficient algorithms and computational concerns

Model means Graphical, mathematical (symbolic), physical, or verbal representation or simplified version of a concept, phenomenon, relationship, structure, system, or an aspect of the real world.

In turn, the belief is the set of principles that helps us to interpret the planning model.
States contain the current states and the next state of the system.
While the actions are the operation required to transfer from one state to another.



Planning Examples

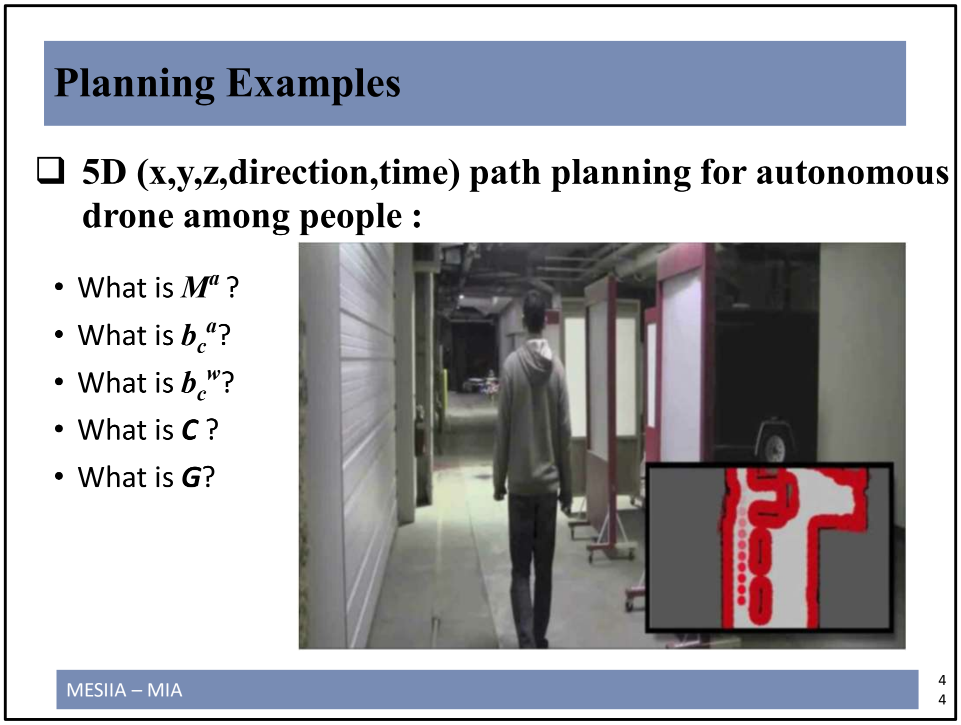
❑ **2D path planning for omnidirectional robot**

- What is model M^a ?
- What is Belief b_c^a?
- What is *Belief* b_c^w?
- What is the cost function ***C*** ? (Path)
- What is the goal ***G***?

Generating a planner is
Given:
A way to describe the world
An initial state of the world
A goal description
A set of possible actions to change the world
Find:
A prescription for actions to change the initial state into one that satisfies the goal

M_a is a 2D path planning to move the robot from the red point to the yellow point in an environment containing a white region that the robot can move inside it and a black region is forbidden to move inside it (b_c^w), so-called **belief**.
The current state is the red point (x1,y1) and the next state is the yellow point (x2,y2) (**goal**). Actions are the methods will used by the robot to move.
For example MOVE_DOWN, MOVE_UP, MOVE_LEFT or MOVE_RIGHT.

Cost Function quantifies the error between predicted values and expected values and presents it in the form of a single real number.
Depending on the problem Cost Function can be formed in many different ways Minimized (**loss**) and Maximized (**reward**).



Planning Examples

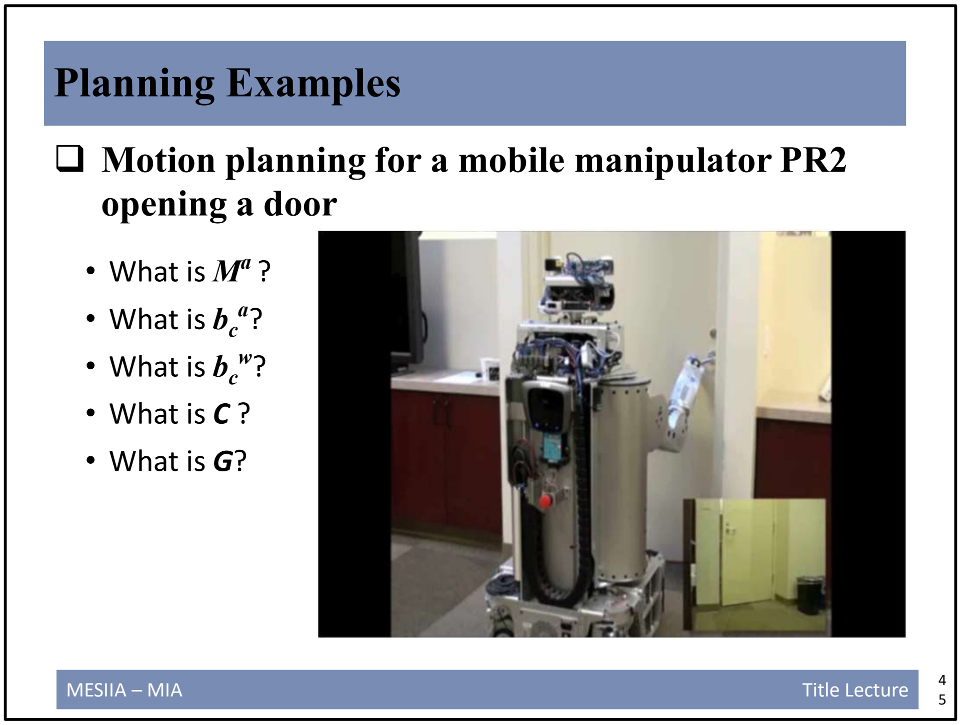
❑ **5D (x,y,z,direction,time) path planning for autonomous drone among people :**

- What is M^a ?
- What is b_c^a?
- What is b_c^w?
- What is C ?
- What is G?

MESIIA – MIA

4
4

It is similar to the pervious problem. It is a **continuous planning system**. Autonomous drone needs to flight indoor (in **5D**) avoiding people (**moving objects**) and walls (**static objects**).



Planning Examples

❑ **Motion planning for a mobile manipulator PR2 opening a door**

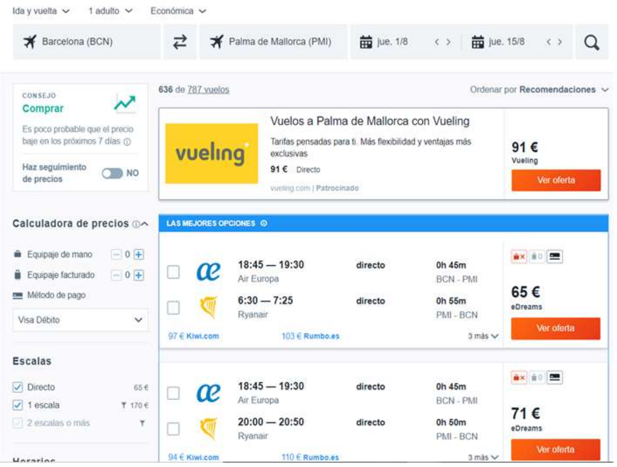
- What is $\boldsymbol{M^a}$?
- What is $\boldsymbol{b_c^a}$?
- What is $\boldsymbol{b_c^w}$?
- What is $\boldsymbol{C}$?
- What is $\boldsymbol{G}$?

Here the robot needs to open the door. So the problem is divided into two problems.
The first is how the robot can reach the door. That is similar to the pervious problem that define how to go from a point in 2D (3D) to another.
The second problem is to manipulate (**catch it**) the knob of the door.
So here we have a **hybrid planning problem**.

Planning Examples

❑ **Planning a travel from Barcelona to Palma Mallorca**

- What is M^a ?
- What is b_c^a?
- What is b_c^w?
- What is ***C*** ?
- What is ***G***?



Ida y vuelta ⌄ 1 adulto ⌄ Económica ⌄

Barcelona (BCN) ⇄ Palma de Mallorca (PMI) jue. 1/8 ‹ › jue. 15/8 ‹ ›

CONSEJO
Comprar
Es poco probable que el precio baje en los próximos 7 días
Haz seguimiento de precios NO

636 de 787 vuelos

Ordenar por **Recomendaciones**

Vuelos a Palma de Mallorca con Vueling
Tarifas pensadas para ti. Más flexibilidad y ventajas más exclusivas
91 € Directo
vueling.com | Patrocinado
91 € Vueling
Ver oferta

Calculadora de precios
Equipaje de mano 0
Equipaje facturado 0
Método de pago
Visa Débito

LAS MEJORES OPCIONES

□	**18:45 — 19:30** Air Europa	**directo**	**0h 45m** BCN - PMI	**65 €** eDreams Ver oferta
□	**6:30 — 7:25** Ryanair	**directo**	**0h 55m** PMI - BCN	
	97 € Kiwi.com	103 € Rumbo.es	3 más ⌄	
□	**18:45 — 19:30** Air Europa	**directo**	**0h 45m** BCN - PMI	**71 €** eDreams Ver oferta
□	**20:00 — 20:50** Ryanair	**directo**	**0h 50m** PMI - BCN	
	94 € Kiwi.com	110 € Rumbo.es	3 más ⌄	

Escalas

☑ Directo	65 €
☑ 1 escala	170 €
□ 2 escalas o más	

Horarios

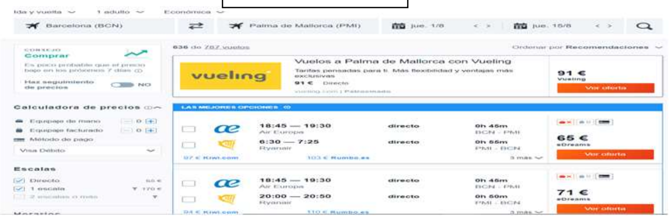
Searching to find the best flight from Barcelona to Palma is **a discrete planning problem**.
As the system needs to go from a point and a transit point and then continue to the target point.
Also, you find different paths and different combination of paths.

Continuous vs. Discrete vs. Hybrid Model

Continuous

Hybrid

Discrete



Ida y vuelta ˅ 1 adulto ˅ Económica ˅

Barcelona (BCN) ⇄ Palma de Mallorca (PMI) jue. 1/8 < > jue. 15/8 < >

CONSEJO
Comprar
Es poco probable que el precio baje en los próximos 7 días
Haz seguimiento de precios NO

636 de 767 vuelos Ordenar por Recomendaciones ˅

Vuelos a Palma de Mallorca con Vueling
Tarifas pensadas para ti. Más flexibilidad y ventajas más exclusivas
91 € Directo
vueling.com | Patrocinado
91 € Vueling Ver oferta

Calculadora de precios
Equipaje de mano 0
Equipaje facturado 0
Método de pago
Visa Débito ˅

LAS MEJORES OPCIONES

□	18:45 — 19:30 Air Europa	directo	0h 45m BCN - PMI	65 € eDreams Ver oferta
□	6:30 — 7:25 Ryanair	directo	0h 55m PMI - BCN	
97 € Kiwi.com	103 € Rumbo.es		3 más ˅	
□	18:45 — 19:30 Air Europa	directo	0h 45m BCN - PMI	71 € eDreams Ver oferta
□	20:00 — 20:50 Ryanair	directo	0h 50m PMI - BCN	
94 € Kiwi.com	110 € Rumbo.es		3 más ˅	

Escalas
☑ Directo 65 €
☑ 1 escala 170 €
□ 2 escalas o más

As you see, we have three different types of planning that are continuous, hybrid and discrete planning.
Can you give other examples about the three types of planning systems?

End