Planning Representation

Dra. Mª Dolores Rodríguez Moreno

Objectives

Specific Objectives

- Role of logics
- Role of search
- Know main languages

Source

- Stuart Russell & Peter Norvig (2009). Artificial Intelligence: A Modern Approach. (3rd Edition). Ed. Pearsons
- Ghallab, Nau &Traverso (2004). Automated Planning: Theory & Practice. The Morgan Kaufmann Series in Artificial Intelligence

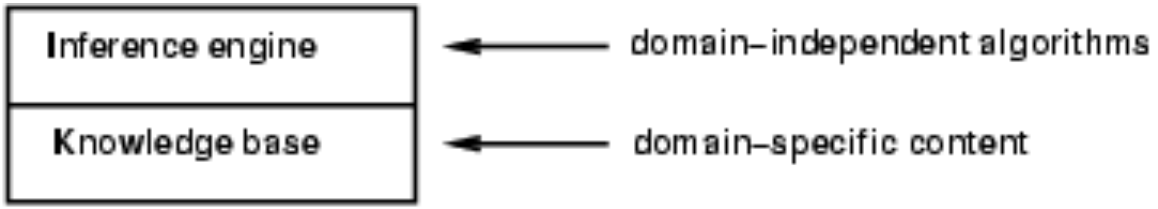
Outline

- **Introduction**
- Logics
- Type of problems
- Modelling in planning
- STRIPS
- PDDL
- IPC
- Conclusions

Introduction

- Knowledge representation and associated processes are central to the AI
- They play an important role when dealing with partially observable environments
  - Combining general knowledge with real perceptions to infer hidden aspects of the world before selecting any action
  - The natural understanding needs also to infer hidden states

Introduction



- Knowledge base = set of sentences in a formal language
- Follow declarative approach to build an agent (or other system):
  - Tell it what it needs to know
  - Then it can ask itself what to do - answers should follow from the KB

Outline

- Introduction
- **Logic**
- Type of problems
- Modelling in planning
- STRIPS
- PDDL
- IPC
- Conclusions

Logic

- Logics are formal languages for representing information such that conclusions can be drawn
- Syntax defines the sentences in the language
- Semantics define the "meaning" of sentences (i.e., define truth of a sentence in a world)
- E.g., the language of arithmetic
  - $x+2 \geq y$ is a sentence; $x2+y > \{\}$ is not a sentence
  - $x+2 \geq y$ is true iff the number $x+2$ is no less than the number y
  - $x+2 \geq y$ is true in a world where $x = 7$, $y = 1$
  - $x+2 \geq y$ is false in a world where $x = 0$, $y = 6$

Logic: types (I)

- Propositional: is the simplest logic and assumes the world contains facts
  - Representation elements: proposition and connectivity ($\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\neg$)
  - Inference: deductions with rules, facts & Modus-Ponens, Modus Tollen
  - Advantage: general representation and decidable in finite time if CNF (DPLL or WalkSAT)
  - Disadvantage: low expressivity and not able to reason about sets of things (i.e. graphs or hierarchies)
- First Order Logic:
  - Based on predicates:
    - Objects: people, houses, numbers, colours, baseball games, wars, …
    - Relations: red, round, prime, brother of, bigger than, part of, comes between, …
    - Functions: father of, best friend, one more than, plus, …
  - Representation elements: predicates, connectivity ($\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, $\neg$) and quantifiers ($\forall$, $\exists$)
  - Inference and Unification

Logic: types (II)

- Second order (or higher order) logic
  - They have two (or three) defined types: objects and sets or functions on them (or both)
  - It is equivalent to saying that predicates can take other predicates as arguments
- Modal and temporal logic
  - It deals with statements affected by necessarily and possibly
- Diffuse logic
  - Quantify the uncertainty
- Others: multi-valued, non-monotonous, quantic,

Outline

- Introduction
- Logic
- **Type of problems**
- Modelling in planning
- STRIPS
- PDDL
- IPC
- Conclusions

Type of problems

- **Fully observable** (vs. partially observable): an agent's sensors give it access to the complete state of the environment at each point in time
- **Deterministic** (vs. non-deterministic): the next state of the environment is completely determined by the current state and the action executed by the agent
- **Episodic** (vs. sequential): the agent's experience is divided into atomic "episodes" (each episode consists of the agent perceiving and then performing a single action), and the choice of action in each episode depends only on the episode itself
- **Discrete** (vs. continuous): A limited number of distinct, clearly defined percepts and actions
- **Single agent** (vs. multiagent): an agent operating by itself in an environment

Outline

- Introduction
- Logic
- Type of problems
- **Modelling in planning**
- STRIPS
- PDDL
- IPC
- Conclusions

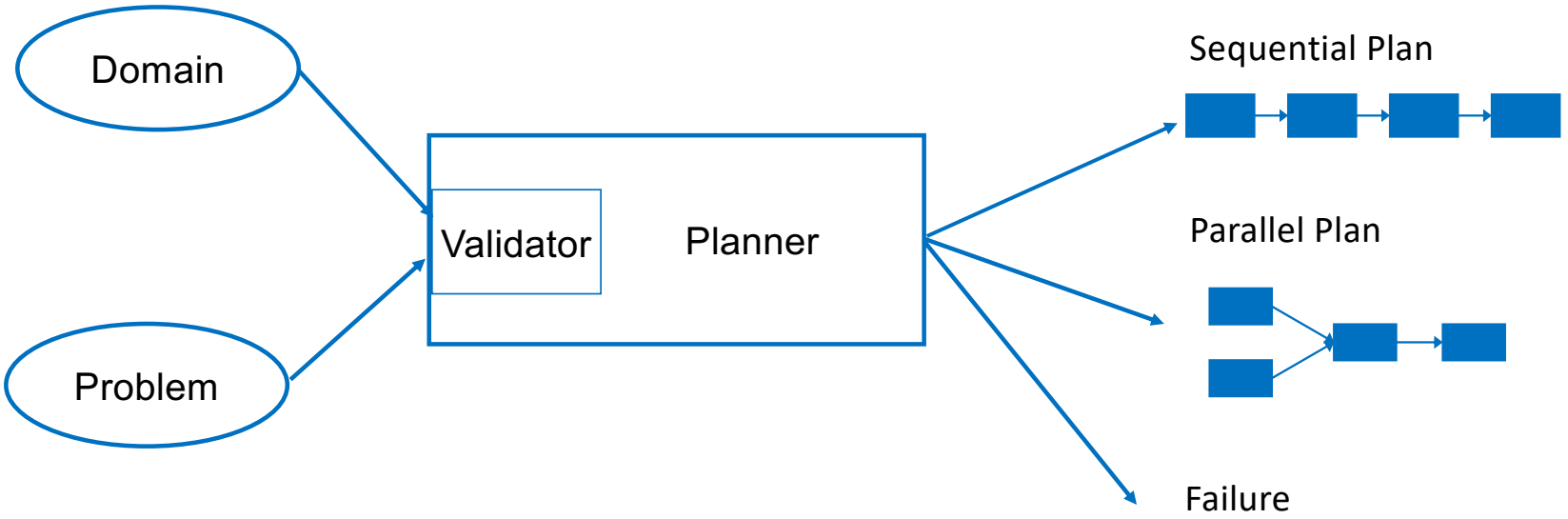
Modelling in planning (I)

- Planners decompose the world in terms of logical conditions and represent a state as a sequence of connected positive literals
  - Propositions: Fast $\wedge$ AirbusA380
  - FOL: At(Plane1,Madrid) $\wedge$ At(Plane2,Paris)
  - Not allowed: At(Padre(Ana),Madrid)
- Representations are based on predicates and objects
- Each domain has a specific FOL containing predicates and functions
- We define a set of operators that are a parametrized representation of the available transitions domain
- It is assumed that all conditions that are not mentioned in a state are FALSE (**closed world assumption**)

Modelling in planning (II)

- Given a classical planning problem
  - Inputs
    - Problem: Initial State and Goal(s)
    - Domain
  - Output
    - A sequence of actions that transform the initial state to a state that satisfies the goals
- State space
  - All states reachable by applying a sequence of actions to the initial state
  - Actions are operators with its parameters instantiated by constants of the initial state

Modelling in planning (III)



Modelling in planning: Domain (I)

- Operators specify the possible transactions in a domain
- For each problem, use the initial state and the specification of operators to determine possible transactions for the particular problem
- To be problem independent, operators use parameters
- Composed of:
  - Preconditions: conditions that must be met before execution
  - Effects: conditions that we achieve when the action is executed
- What is not mentioned, remains unchanged

Modelling in planning: Domain (II)

- We say that an action is applicable in any state that satisfies its preconditions, otherwise the action has no effect
- The application consists of replacing the variables by constants
- Instantiated operator is called an action

Modelling in planning: Problem

- Composed of Initial State and Goal(s)
- To represent initial state
  - It can be any logical description
  - What is not mentioned is FALSE
- To represent a goal
  - A set of literals (*ground literals*)
  - A goal is satisfied if **all** literals are satisfied

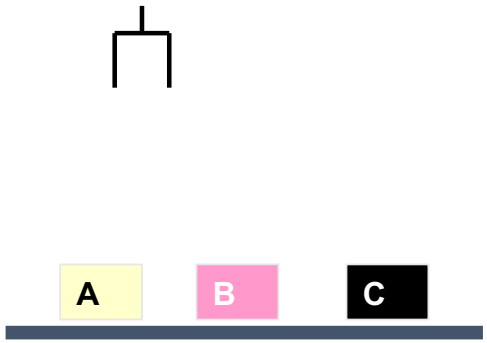
Outline

- Introduction
- Logic
- Type of problems
- Modelling in planning
- **STRIPS**
- PDDL
- IPC
- Conclusions

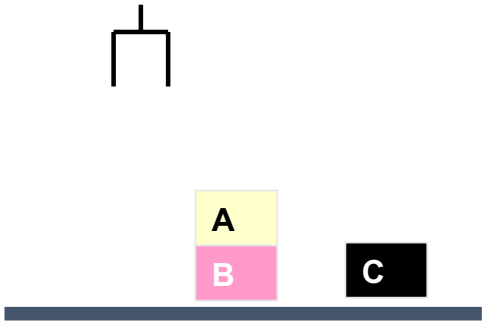
STRIPS

- **St**anford **R**esearch **I**nstitute **P**roblem **S**olver
- It is an automated planner and a formal language
- Simplest and oldest representation of operators in AI (1971)
- The initial state is represented by a DB of positive facts
- It can be seen as a simple way to specify updates to the DB
- Little expressiveness for real domains so that other languages have emerged: ADL, Prodigy language (PDL), UCPOP language , SIPE-2 language ,..., PDDL (standard)

STRIPS (I)



Initial State



Goal

STRIPS (II)

```
(def-strips-operator (pickup ?x)
   (pre (handempty) (clear ?x) (ontable ?x))
   (add (holding ?x))
   (del (handempty) (clear ?x) (ontable ?x)))
```

STRIPS (II)

```
(def-strips-operator (pickup ?x)
    (pre (handempty) (clear ?x) (ontable ?x))
    (add (holding ?x))
    (del (handempty) (clear ?x) (ontable ?x)))
```

Operator name & parameters

STRIPS (II)

```
(def-strips-operator (pickup ?x)
 (pre (handempty) (clear ?x) (ontable ?x))
List of predicates that must be true
in the current state for aplying the
action
 (add (holding ?x))
 (del (handempty) (clear ?x) (ontable ?x)))
```

STRIPS (II)

```
(def-strips-operator (pickup ?x)
   (pre (handempty) (clear ?x) (ontable ?x))
   (add (holding ?x))
   List of predicates that must be true
   in the next state

   (del (handempty) (clear ?x) (ontable ?x)))
```

STRIPS (II)

```
(def-strips-operator (pickup ?x)
   (pre (handempty) (clear ?x) (ontable ?x))
   (add (holding ?x))
   (del (handempty) (clear ?x) (ontable ?x)))
   List of predicates that must be false
   in the next state
```

STRIPS (III)

- Given the initial state
  - All instantiations of the `?x` parameter that satisfy the precondition

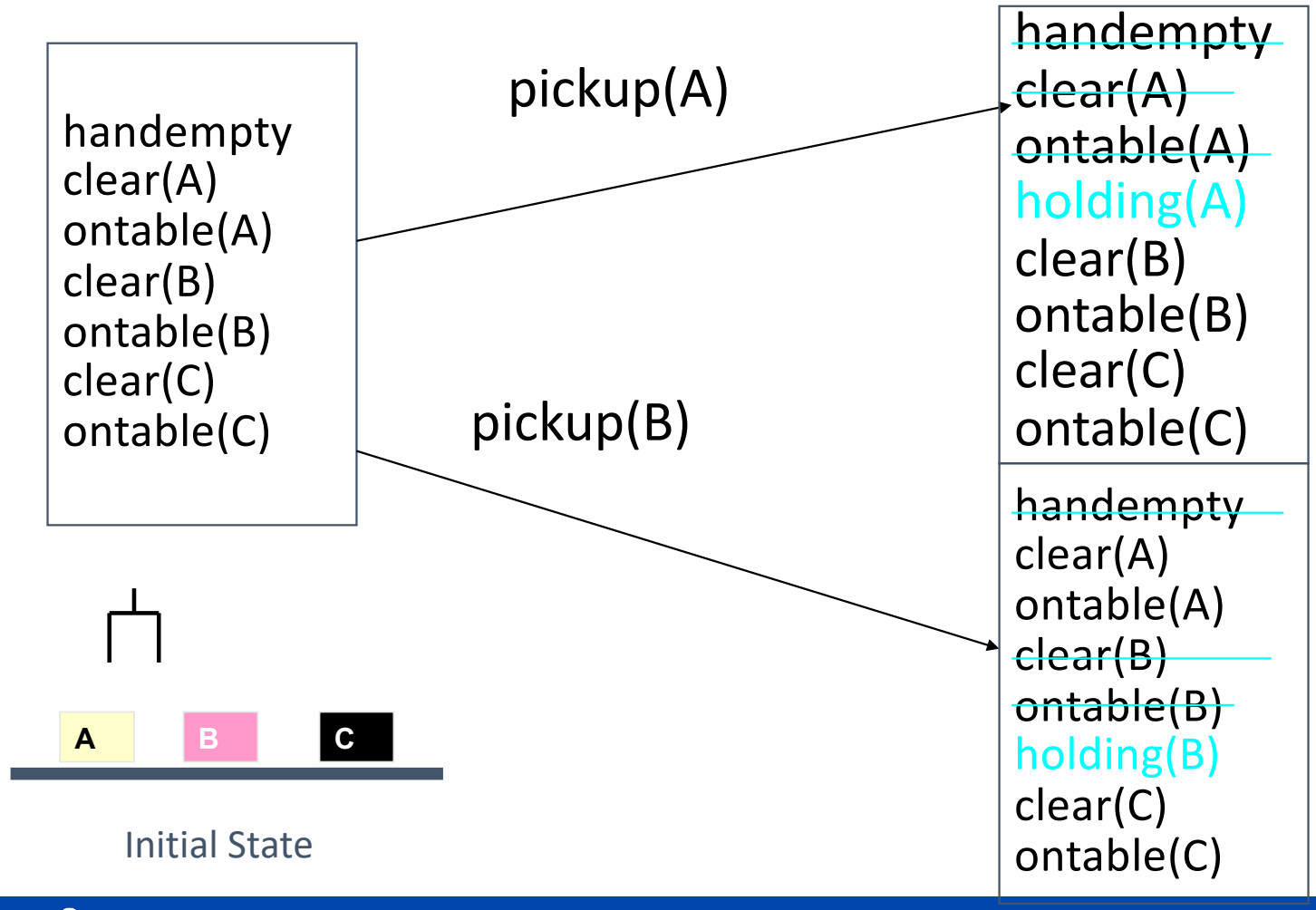
```
(and (handempty) (clear ?x) (ontable ?x))
```

    produced a different action (transition) that can be applied to the initial state
  - Actions whose preconditions are not met are illegal transitions

STRIPS (IV)

- Actions are deterministic
- Nothing else changes more! (This often has algorithmic consequences)
- From the properties of the initial state and the set of operators is possible to determine:
  - The finite set of actions that can be applied to the initial state
  - In each successor state generated by these actions, you can evaluate all logical formulas, and determine the set of available actions

STRIPS



Initial State

Outline

- Introduction
- Logic
- Type of problems
- Modelling in planning
- STRIPS
- **PDDL**
- IPC
- Conclusions

PDDL

- **P**lanning **D**omain **D**efinition **L**anguage
- Emerges as an attempt to unify previous formalisms and to compare the efficiency of planners
- AIPS-98 Planning Competition Committee
- Intended to represent the physics of a domain: what predicates there are, what actions are possible, the structure of such actions and their effects
- Based on ADL, Prodigy language (PDL), UCPOP language, among others
- Versions: 1.2, 2.1, 3.0 and actually in the 3.1

PDDL2.1 Variants

- **PDDL+:** provides a more flexible model of continuous changes through the use of autonomous processes and events
- **MAPL** (Multi-Agent Planning Language, pronounced "maple")
  - Non-propositional state-variables (which may be n-ary: true, false, unknown, or anything else)
  - Temporal model given with modal operators (before, after, etc.).
  - Actions whose duration will be determined in runtime and explicit plan synchronization which is realized through speech act based communication among agents

(Source)

PDDL2.1 Variants

- **OPT** (Ontology with Polymorphic Types):
  - An attempt to create a general-purpose notation for creating ontologies, defined as formalized conceptual frameworks for planning domains
  - Efficient type inference and other compatibilities with the semantic web
- **PPDDL** (Probabilistic PDDL):
  - Probabilistic effects (discrete, general probability distributions over possible effects of an action)
  - Reward fluents (for incrementing or decrementing the total reward of a plan in the effects of the actions)
  - Goal rewards (for rewarding a state-trajectory, which incorporates at least one goal-state)
  - Goal-achieved fluents (which were true, if the state-trajectory incorporated at least one goal-state)

Variants

- **RDDL** (Relational Dynamic influence Diagram Language): the 7th IPC in 2011
  - Based on PPDDL1.0 and PDDL3.0, is a completely different language both syntactically and semantically
  - Introduces partial observability (allows efficient description of MDPs and POMDPs by representing everything with variables
- **NDDL** (New Domain Definition Language) is NASA's planning language
  - Use variable representation (timelines/activities) rather than a propositional/first-order logic, and
  - No concept of states or actions, only of intervals (activities) and constraints between activities

Outline

- Introduction
- Logic
- Type of problems
- Modelling in planning
- STRIPS
- PDDL
- **IPC**
- Conclusions

IPC

- **I**nternational **P**lanning **C**ompetition
- Organized in the context of the International Conference on Planning and Scheduling (ICAPS)
- Aims to evaluate SoA planning systems on a number of benchmark problems
- Goals:
  - Promote planning research
  - Highlight challenges in the planning community
  - Provide new and interesting problems as benchmarks for future research

IPC: competitions

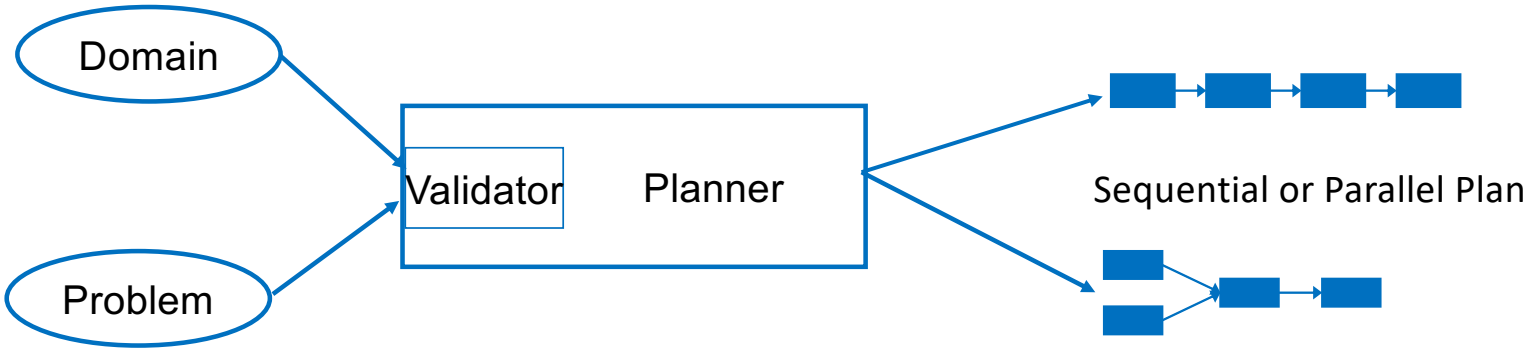
- 1st IPC in 1998 (1 track: Deterministic)
- 2nd IPC in 2000 (1 track: Deterministic)
- 3rd IPC in 2002 (1 track: Deterministic)
- 4th IPC in 2004, University of Freiburg, Germany (2 tracks: Deterministic & Probabilistic)
- 5th IPC in 2006, Universita di Brescia, Italy (2 tracks: Deterministic & Probabilistic)
- 6th IPC in 2008 (3 tracks: Deterministic, Uncertainty & Learning)
- 7th IPC in 2011 (3 tracks: Deterministic, Uncertainty & Learning)
- 8th IPC in 2014, University of Huddersfield (4 tracks: Deterministic, Uncertainty & Learning)
- 9th IPC in 2018 (3 tracks: Deterministic, Probabilistic & Temporal)

Outline

- Introduction
- Logic
- Type of problems
- Modelling in planning
- STRIPS
- PDDL
- IPC
- **Conclusions**

Conclusions

- Knowledge representation in FOL
- Languages: STRIPS and PDDL (standard)
- Competitions: IPC
- Inputs & outputs of a planner



Conclusions