Principles of AI Planning

January 12th, 2007 — Nondeterministic planning

Principles of AI Planning

Nondeterministic planning

Malte Helmert Bernhard Nebel

Albert-Ludwigs-Universität Freiburg

January 12th, 2007

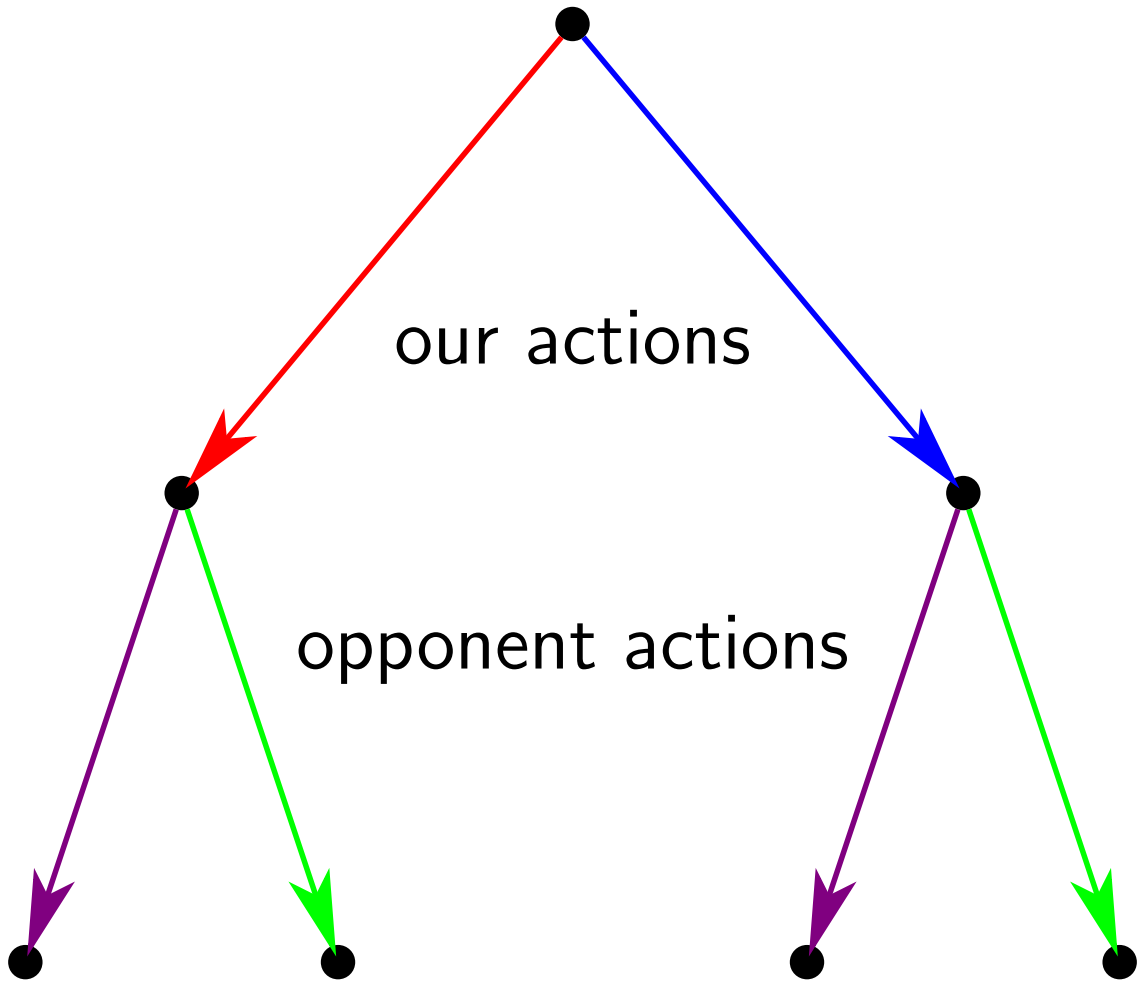
Nondeterministic planning

Motivation

- The world is not predictable.
- AI robotics:
  - imprecise movement of the robot
  - other robots
  - human beings, animals
  - machines (cars, trains, airplanes, lawn-mowers, ...)
  - natural phenomena (wind, water, snow, temperature, ...)
- Games: other players are outside our control.
  - To win a game (reaching a goal state) with certainty, all possible actions by the other players have to be anticipated (a winning strategy of a game).
  - The world is not predictable because it is unknown: we cannot observe everything.

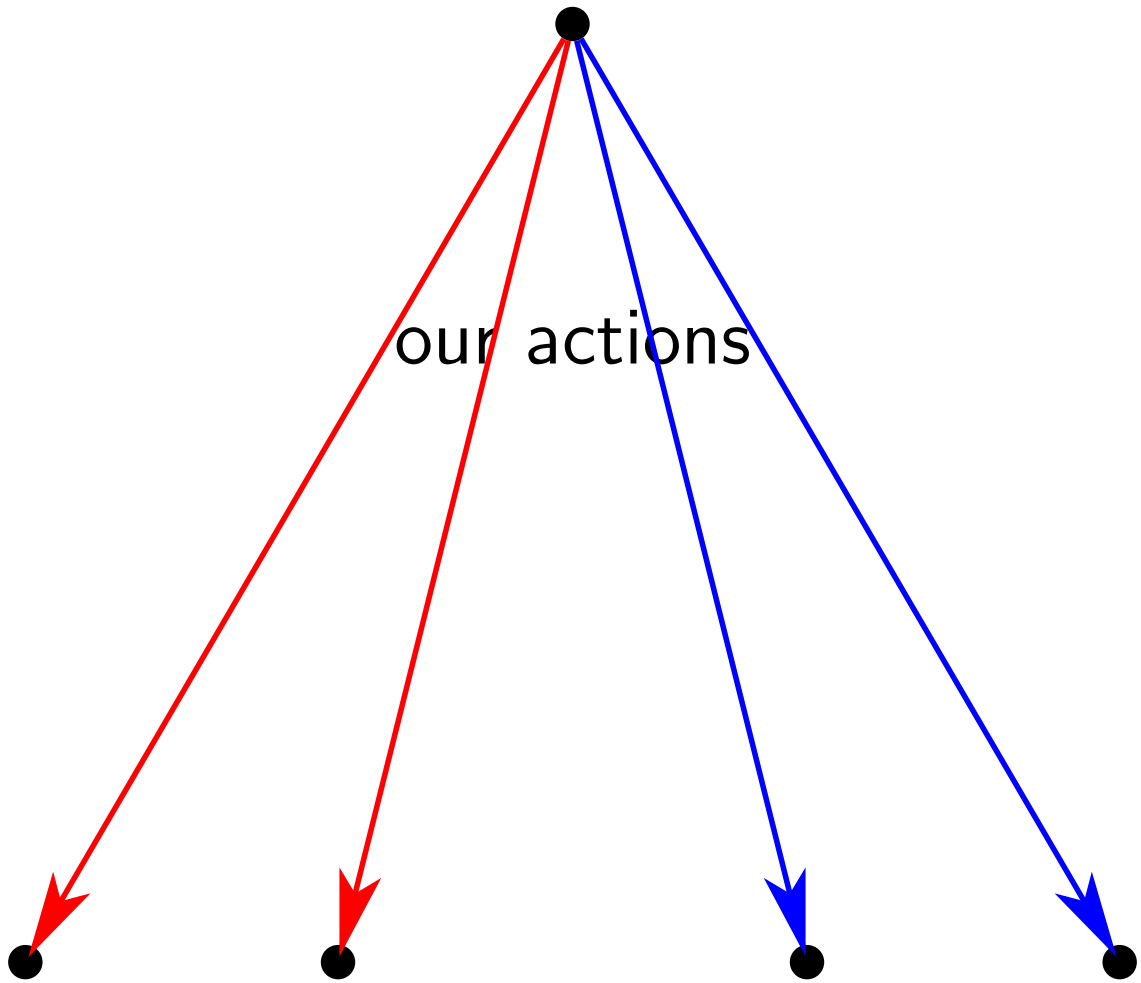
Nondeterminism

Example: several agents, games



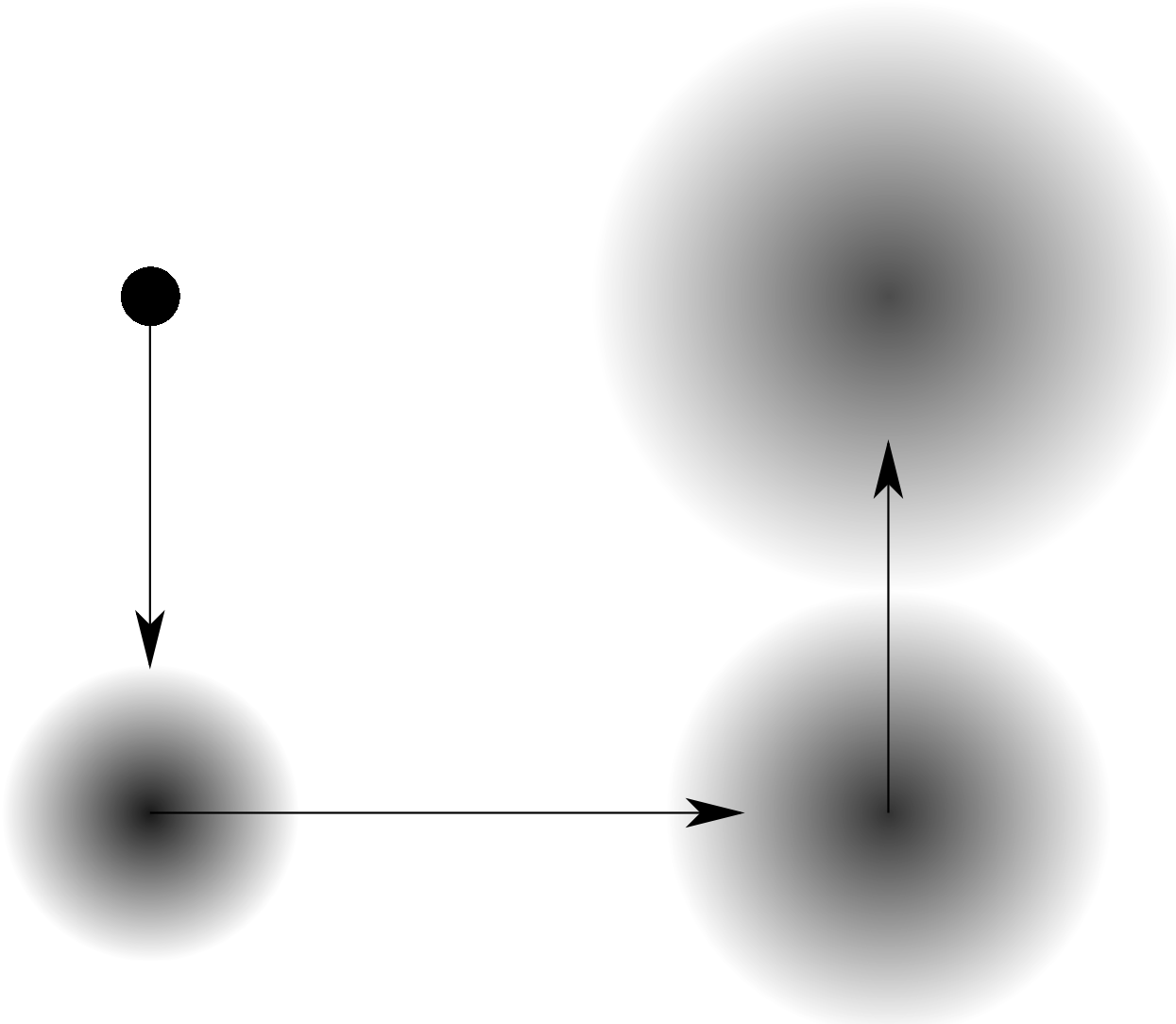
Nondeterminism

Example: several agents, games



Nondeterminism

Example: uncertainty in robot movement



Nondeterministic planning

Motivation

- In deterministic planning we have assumed that the only changes taking place in the world are those caused by us and that we can exactly predict the results of our actions.
- Other agents and processes, beyond our control, are formalized as nondeterminism.
- Implications:
  1. The future state of the world cannot be predicted.
  2. We cannot reliably plan ahead: no single action sequence achieves the goals.
  3. In some cases it is not possible to achieve the goals with certainty, only with some probability.

Transition systems

General definition with nondeterminism and observability

Definition

A transition system is a 5-tuple $\Pi = \langle S, I, O, G, P \rangle$ where

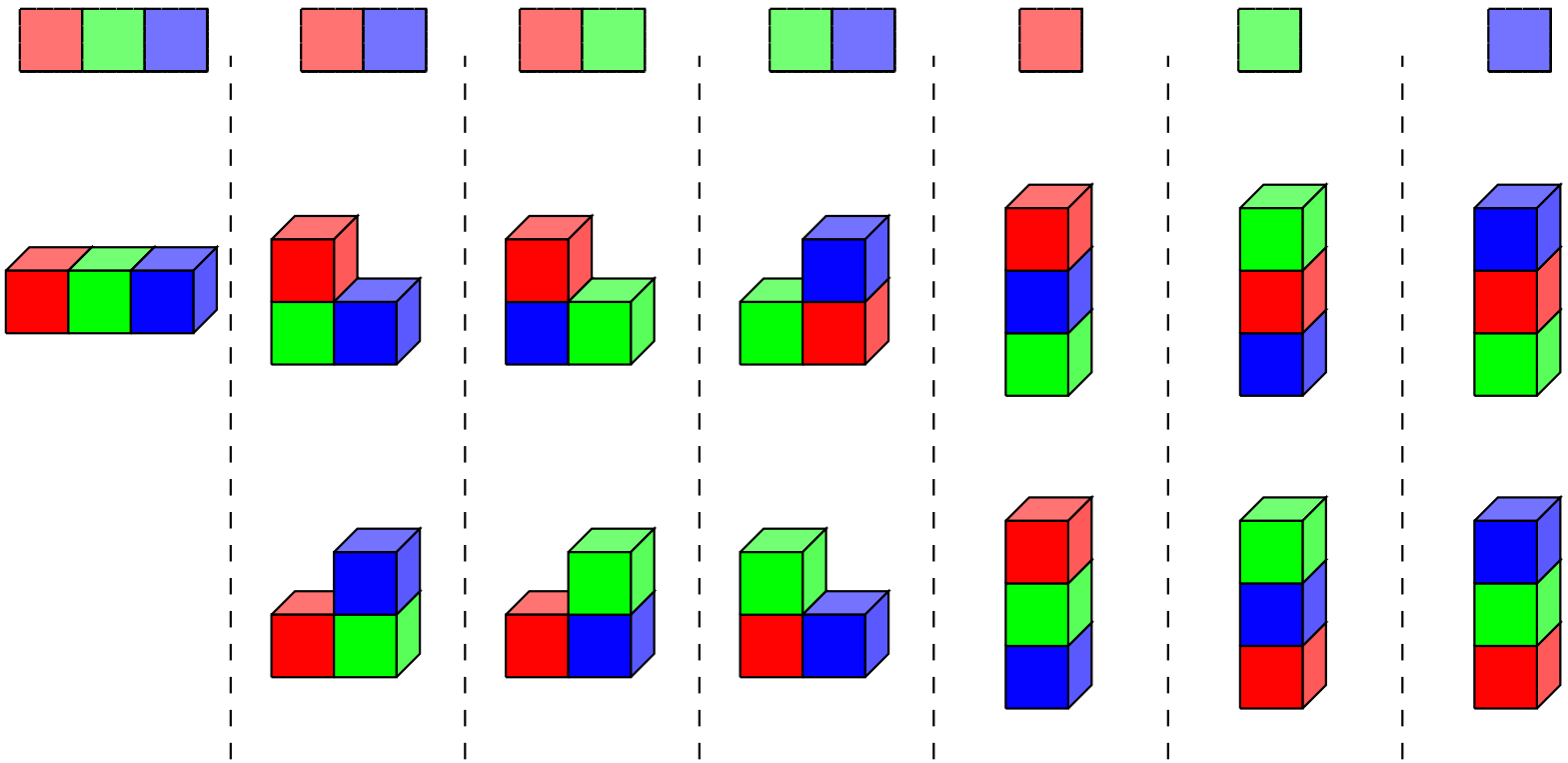
1. S is a finite set of states,
2. $I \subseteq S$ is the set of initial states,
3. O is a finite set of actions $o \subseteq S \times S$,
4. $G \subseteq S$ is the set of goal states, and
5. $P = (C_1, \ldots, C_n)$ is a partition of S to classes of observationally indistinguishable states satisfying $\bigcup\{C_1, \ldots, C_n\} = S$ and $C_i \cap C_j = \emptyset$ for all i, j such that $1 \leq i < j \leq n$.

Making an observation tells which set C_i the current state belongs to.
Distinguishing states within a given C_i is not possible by observations.

Observability

Example of partition of states into observational classes

Blocks world with 3 blocks and camera far above the table.



Observability

Classification full, partial, no observability

Let $S = \{s_1, \ldots, s_n\}$ be the set of states.
Classification of planning problems in terms of observability:

Full $P = (\{s_1\}, \{s_2\}, \ldots, \{s_n\})$
number of observational classes: n
Chess is a fully observable 2-person game.

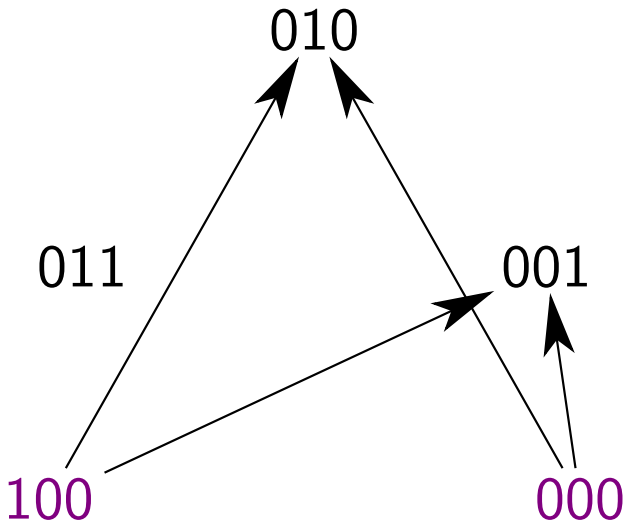
No $P = (\{s_1, \ldots, s_n\})$
number of observational classes: 1

Partial No restrictions on P.
number of observational classes: between 1 and n
Poker is a partially observable k-person game.
Mastermind is a partially observable 1-person game.

n-person games for $n \geq 2 \sim$ nondeterministic planning

Nondeterministic actions as operators

Example



	000	001	010	011	100	101	110	111
000	0	1	1	0	0	0	0	0
001	0	0	0	0	0	0	0	0
010	0	0	0	0	0	0	0	0
011	0	0	0	0	0	0	0	0
100	0	1	1	0	0	0	0	0
101	0	0	0	0	0	0	0	0
110	0	0	0	0	0	0	0	0
111	0	0	0	0	0	0	0	0

In terms of state variables $A = \{a, b, c\}$ the action can be represented as operator

$$\langle \neg b \wedge \neg c, \neg a \wedge (b|c) \rangle$$

Nondeterministic actions as operators

Definition

Definition

Let A be a set of state variables. An operator is a pair $\langle c, e \rangle$ where c is a propositional formula over A (the precondition), and e is an effect over A. Effects over A are recursively defined as follows.

1. a and $\neg a$ for state variables $a \in A$ are effects over A.
2. $e_1 \wedge \cdots \wedge e_n$ is an effect over A if $e_1, \ldots, e_n$ are effects over A.
3. $c \rhd e$ is an effect over A if c is a formula over A and e is an effect over A.
4. $e_1 | \ldots | e_n$ is an effect over A if $e_1, \ldots, e_n$ for $n \geq 2$ are effects over A.

Nondeterministic operators

Semantics, example

Example

$$\langle a, (b|\neg b) \wedge (c|\neg c) \wedge (d|\neg d) \rangle$$

has 2^3 alternative sets of effects, leading to 8 different successor states.

1. effects $\{b, c, d\}$ lead to state $s \models a \wedge b \wedge c \wedge d$
2. effects $\{\neg b, c, d\}$ lead to state $s \models a \wedge \neg b \wedge c \wedge d$
3. effects $\{b, \neg c, d\}$ lead to state $s \models a \wedge b \wedge \neg c \wedge d$
4. effects $\{\neg b, \neg c, d\}$ lead to state $s \models a \wedge \neg b \wedge \neg c \wedge d$
5. effects $\{b, c, \neg d\}$ lead to state $s \models a \wedge b \wedge c \wedge \neg d$
6. effects $\{\neg b, c, \neg d\}$ lead to state $s \models a \wedge \neg b \wedge c \wedge \neg d$
7. effects $\{b, \neg c, \neg d\}$ lead to state $s \models a \wedge b \wedge \neg c \wedge \neg d$
8. effects $\{\neg b, \neg c, \neg d\}$ lead to state $s \models a \wedge \neg b \wedge \neg c \wedge \neg d$

Nondeterministic operators

Semantics

Definition (Operator application)

Let $\langle c, e \rangle$ be an operator over A and s a state.
The set $[e]_s$ of sets of literals is recursively defined as follows.

1. $[a]_s = \{\{a\}\}$ and $[\neg a]_s = \{\{\neg a\}\}$ for $a \in A$
2. $[e_1 \wedge \cdots \wedge e_n]_s = \{\bigcup_{i=1}^{n} E_i \mid E_1 \in [e_1]_s, \ldots, E_n \in [e_n]_s\}$
3. $[c' \rhd e]_s = [e]_s$ if $s \models c'$ and $[c' \rhd e]_s = \{\emptyset\}$ otherwise
4. $[e_1 | \ldots | e_n]_s = [e_1]_s \cup \cdots \cup [e_n]_s$

Definition

Operator $\langle c, e \rangle$ is applicable in s if $s \models c$ and every set $E \in [e]_s$ is consistent.

Nondeterministic operators

Binary relation induced by an operator

Definition

An operator $\langle c, e \rangle$ induces a binary relation $R\langle c, e \rangle$ on the states as follows: $sR\langle c, e \rangle s'$ if there is $E \in [e]_s$ such that

1. $s \models c$,
2. $s' \models E$, and
3. $s \models a$ iff $s' \models a$ for all $a \in A$ such that $\{a, \neg a\} \cap E = \emptyset$.

We also write simply sos' instead of $sR(o)s'$.

Definition

Let s and s' be states and o an operator. If sos' then s' is a successor state of s.

Succinct transition systems

General definition

Definition

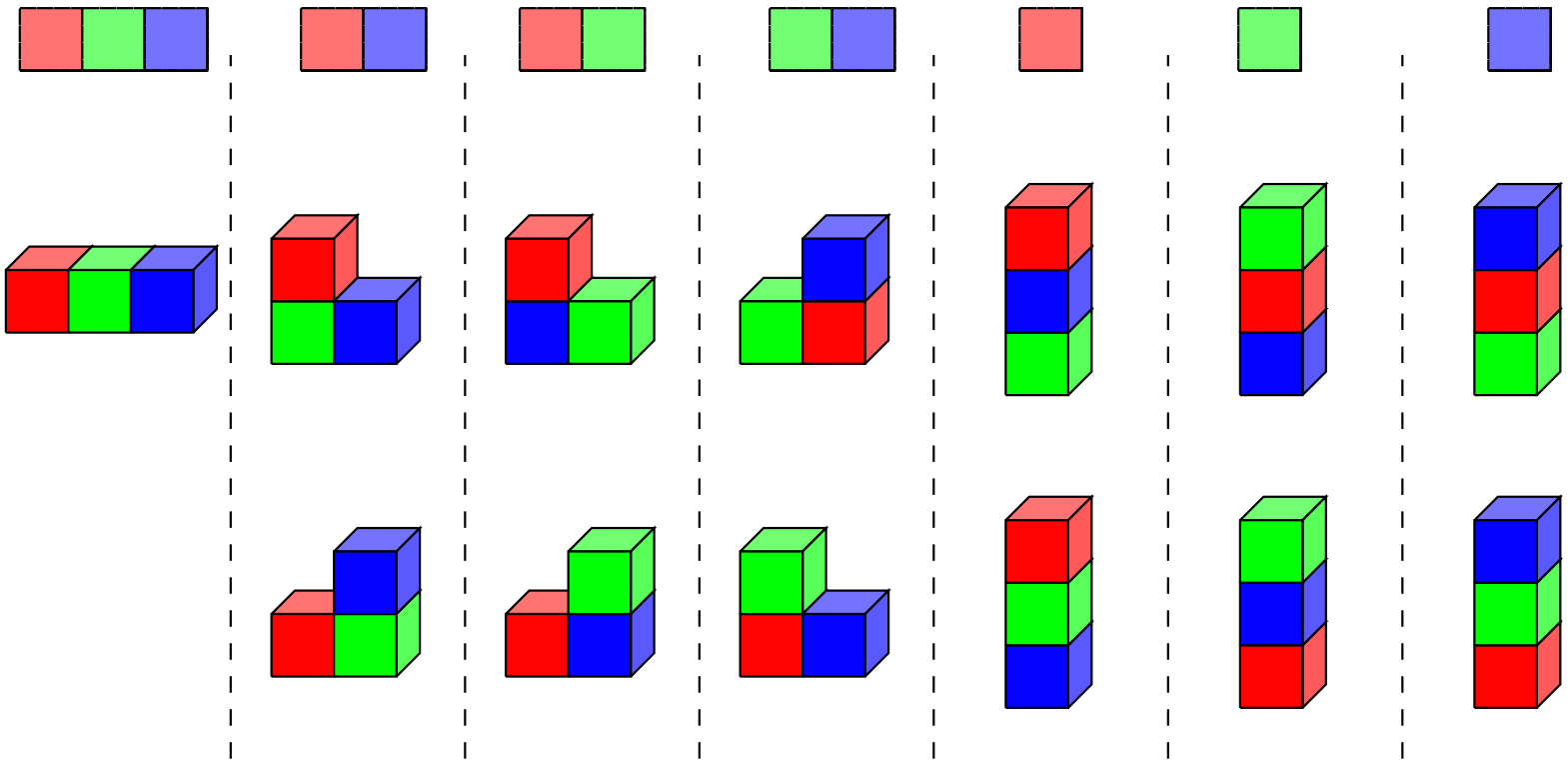
A succinct transition system is a 5-tuple $\Pi = \langle A, I, O, G, V \rangle$ where

1. A is a finite set of state variables,
2. I is a formula over A describing the initial states,
3. O is a finite set of operators over A,
4. G is a formula over A describing the goal states, and
5. $V \subseteq A$ is the set of observable state variables.

Observability

Example of partition of states into observational classes

State variables $V = \{Aclear, Bclear, Cclear\}$ are observable.



There are 8 valuations of V, but the valuation $v \models \neg Aclear \wedge \neg Bclear \wedge \neg Cclear$ does not correspond to a blocks world state.

Succinct transition systems

Observability

Let $A = \{a_1, \ldots, a_n\}$ be the state variables.
Classification of planning problems in terms of observability:

Full observable state variables: $V = A$
number of observational classes: $2^{|A|}$

No observable state variables: $V = \emptyset$
number of observational classes: 1

Partial observable state variables: no restrictions, $\emptyset \subseteq V \subseteq A$
number of observational classes: 1 to $2^{|A|}$

Succinct transition system

Translation into transition systems

We can associate a transition system with every succinct transition system.

Definition

Given a succinct transition system $\Pi = \langle A, I, O, G, V \rangle$, define the transition system $F(\Pi) = \langle S, I', O', G', P \rangle$ where

1. S is the set of all Boolean valuations of A,
2. $I' = \{s \in S \mid s \models I\}$,
3. $O' = \{R(o) \mid o \in O\}$,
4. $G' = \{s \in S \mid s \models G\}$, and
5. $P = (C_1, \ldots, C_n)$ where $v_1, \ldots, v_n$ for $n = 2^{|V|}$ are all the Boolean valuations of V and $C_i = \{s \in S \mid s(a) = v_i(a) \text{ for all } a \in V\}$ for all $i \in \{1, \ldots, n\}$.

What is a plan?

In nondeterministic planning, plans are much more complicated objects than in the deterministic case:

- The best action to take may depend on nondeterministic effects of previous operators, or on the nondeterminism of the initial state.

Nondeterministic plans thus often require branching. Sometimes, they even require looping.

What is a plan?

Example of branching

(Part of) a plan for winning the game Connect Four can be described as follows:

- Place a tile in the 4th column.
  - If opponent places a tile in the 1st, 4th or 7th column, place a tile in the 4th column.
  - If opponent places a tile in the 2nd or 5th column, place a tile in the 2nd column.
  - If opponent places a tile in the 3rd or 6th column, place a tile in the 6th column.

There is no non-branching plan that solves the task (= is guaranteed to win the game).

What is a plan?

Example of looping

A plan for building a card house can be described as follows:

1. Build a wall with two cards.
   If the structure falls apart, redo from start.
2. Build a second wall with two cards.
   If the structure falls apart, redo from start.
3. Build a ceiling on top of the walls with a fifth card.
   If the structure falls apart, redo from start.
4. Build a wall on top of the ceiling with two cards.
   If the structure falls apart, redo from start.

There is no non-looping plan that solves the task
(unless the planning agent is very dextrous).

What is a plan?

Branching and looping: Intuition

- Plans should be allowed to branch. Otherwise, most interesting nondeterministic planning tasks cannot be solved.
- We may or may not allow plans to loop.
  - Non-looping plans are preferable because they guarantee that the goal is reached within a bounded number of steps.
  - Where non-looping plans are not possible, looping plans may be adequate because they at least guarantee that the goal will be reached eventually unless nature is unfair.
    - While this sounds quite informal, there are in fact formal notions of "fairness" defined for nondeterministic transition systems which are applicable here.

We will now introduce the formal concepts necessary to define branching and looping plans.

Nondeterministic plans: Formal definition

Belief states

Definition (belief state)

Let $\mathcal{T}$ be a nondeterministic planning task with states S.
A belief state $b \subseteq S$ of $\mathcal{T}$ is a set of states of $\mathcal{T}$.

- The name "belief state" relates to the intuition that at any point of plan execution, we only know ("believe") that we are in one state from a certain set of states, but we do not know which one (due to limited observability).

Nondeterministic plans: Formal definition

Applying operators in belief states

Definition (applying operators in belief states)

Let b be a belief state and o be an operator of a nondeterministic planning task with state set S.
The operator o is applicable in belief state b iff it is applicable in each state $s \in b$.
The result of applying o in b is the belief state
$app_o(b) := \{s' \in S \mid \exists s \in b : sos'\}$.

Nondeterministic plans: Formal definition

Strategies

Definition

Let $\mathcal{T} = \langle A, I, O, G, V \rangle$ be a nondeterministic planning task (succinct transition system) with state set S.

A strategy for $\mathcal{T}$ is a directed graph π with labeled vertices and arcs and the following properties:

- Each vertex n is labeled with a belief state $b(n)$ of S.
- One vertex, called the initial node n_0, is labeled with the initial belief state $b(n_0) = \{s \in S \mid s \models I\}$.
- All vertices are reachable from the initial node.
- Each vertex has either 0, 1, or 2 successors.
- ...

Nondeterministic plans: Formal definition

Strategies

Definition (ctd.)

- A vertex n with 0 successors is a leaf node.
- A vertex n with 1 successor is an operator node.
  - Its outgoing arc is labeled with an operator $o \in O$ applicable in $b(n)$.
  - The successor n' satisfies $b(n') = app_o(b(n))$.
- A vertex n with 2 successors is an observation node.
  - Its outgoing arcs are labeled with φ and $\neg\varphi$ for some formula φ over V.
  - The successor n' reached via the arc with label ψ satisfies $b(n') = \{s \in b(n) \mid s \models \psi\}$.

Nondeterministic plans: Formal definition

Strategies vs. applicable operator sequences

- Strategies in nondeterministic planning correspond to applicable operator sequences in deterministic planning.
- In deterministic planning, a plan is an applicable operator sequence that results in a goal state.
- In nondeterministic planning, we define different notions of "resulting in a goal state".

Nondeterministic plans: Formal definition

Plans

Definition

Let $b_\star$ be the set of goal states of a nondeterministic planning task $\mathcal{T}$.

- A strategy is called a weak plan for $\mathcal{T}$
  iff there is a leaf node n such that $b(n) \cap b_\star \neq \emptyset$.
- A strategy for $\mathcal{T}$ is called a strong cyclic plan for $\mathcal{T}$
  iff from each node, a leaf node is reachable, and $b(n) \subseteq b_\star$ for all leaf nodes n.
- A strong cyclic plan for $\mathcal{T}$ is called a strong plan for $\mathcal{T}$
  iff it contains no cycles.
- A strong plan for $\mathcal{T}$ is called a conformant plan for $\mathcal{T}$
  iff it contains no observation nodes.

Nondeterministic plans: Formal definition

Plans vs. observability

There is a relationship between plan properties and the observability class of a planning task:

- For fully observable planning tasks, we can always insert sufficiently many observation nodes before each operator node n to reduce the belief state $b(n)$ to a singleton.
  Moreover, we can assume that no two nodes are labeled with the same belief states.
  - Such plans can be equivalently represented as memoryless plans (also called state-action tables or policies): partial functions $\pi : S \rightarrow O$ which map each state s reachable within the plan to an action $\pi(s)$ to execute in this state.
- For unobservable planning tasks, any (non-weak) plan is conformant (ignoring "useless" observations like $\top \wedge \neg\bot$).
  - Such plans can be equivalently represented as operator sequences.

Summary and outlook

Nondeterministic planning: new concepts

We extended the deterministic (classical) planning formalism in two major ways:

- initial states and operators can be nondeterministic
- observability can be limited

As a consequence, plans can contain

- branches and
- loops.

In the following lectures, we consider different variants of the nondeterministic planning problem, discuss some algorithms and consider computational complexity.