

# Planning and Optimization

## D2. Abstractions: Formal Definition and Heuristics

Malte Helmert and Gabriele Röger

Universität Basel

November 2, 2020

# Planning and Optimization

November 2, 2020 — D2. Abstractions: Formal Definition and Heuristics

D2.1 Reminder: Transition Systems

D2.2 Abstractions

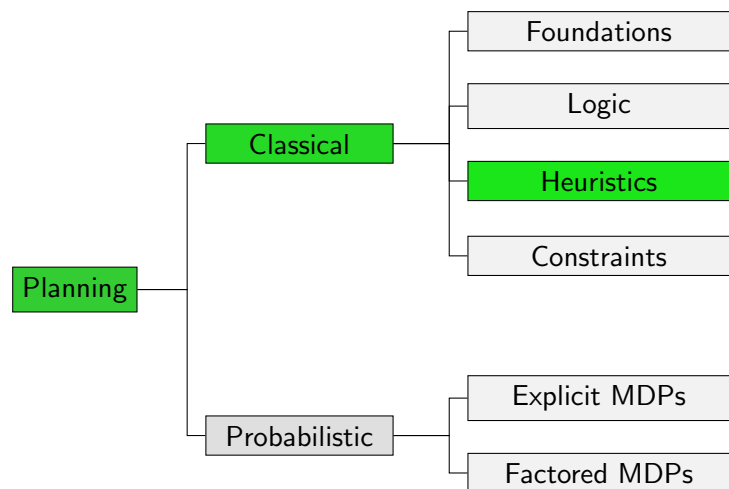
D2.3 Homomorphisms and Isomorphisms

D2.4 Abstraction Heuristics

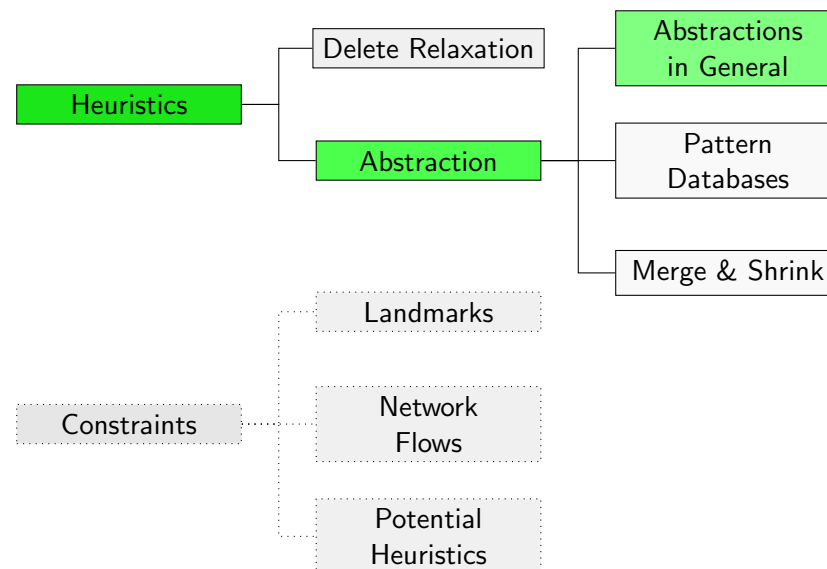
D2.5 Coarsenings and Refinements

D2.6 Summary

## Content of this Course



## Content of this Course: Heuristics



## D2.1 Reminder: Transition Systems

## Transition Systems

Reminder from Chapter A3:

### Definition (Transition System)

A **transition system** is a 6-tuple  $\mathcal{T} = \langle S, L, c, T, s_0, S_* \rangle$  where

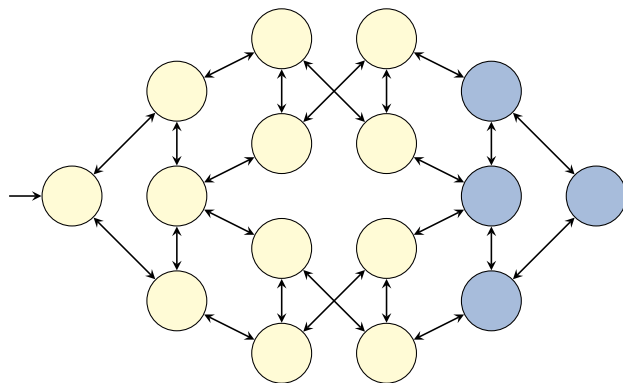
- ▶  $S$  is a finite set of **states**,
- ▶  $L$  is a finite set of (transition) **labels**,
- ▶  $c : L \rightarrow \mathbb{R}_0^+$  is a **label cost** function,
- ▶  $T \subseteq S \times L \times S$  is the **transition relation**,
- ▶  $s_0 \in S$  is the **initial state**, and
- ▶  $S_* \subseteq S$  is the set of **goal states**.

We say that  $\mathcal{T}$  **has the transition**  $\langle s, \ell, s' \rangle$  if  $\langle s, \ell, s' \rangle \in T$ .

We also write this as  $s \xrightarrow{\ell} s'$ , or  $s \rightarrow s'$  when not interested in  $\ell$ .

**Note:** Transition systems are also called **state spaces**.

## Transition Systems: Example



**Note:** To reduce clutter, our figures often omit arc labels and costs and collapse transitions between identical states. However, these are important for the formal definition of the transition system.

## Mapping Planning Tasks to Transition Systems

Reminder from Chapter A3:

### Definition (Transition System Induced by a Planning Task)

The planning task  $\Pi = \langle V, I, O, \gamma \rangle$  **induces** the transition system  $\mathcal{T}(\Pi) = \langle S, L, c, T, s_0, S_* \rangle$ , where

- ▶  $S$  is the set of all states over state variables  $V$ ,
- ▶  $L$  is the set of operators  $O$ ,
- ▶  $c(o) = \text{cost}(o)$  for all operators  $o \in O$ ,
- ▶  $T = \{ \langle s, o, s' \rangle \mid s \in S, o \text{ applicable in } s, s' = s[o] \}$ ,
- ▶  $s_0 = I$ , and
- ▶  $S_* = \{ s \in S \mid s \models \gamma \}$ .

## Tasks in Finite-Domain Representation

### Notes:

- ▶ We will focus on planning tasks in **finite-domain representation** (FDR) while studying abstractions.
- ▶ All concepts apply equally to propositional planning tasks.
- ▶ However, FDR tasks are almost always used by algorithms in this context because they tend to have fewer **useless** (physically impossible) states.
- ▶ Useless states can hurt the efficiency of abstraction-based algorithms.

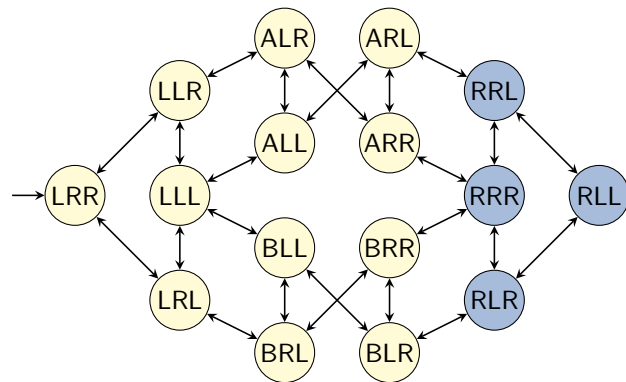
## Example Task: One Package, Two Trucks

### Example (One Package, Two Trucks)

Consider the following FDR planning task  $\langle V, I, O, \gamma \rangle$ :

- ▶  $V = \{p, t_A, t_B\}$  with
  - ▶  $\text{dom}(p) = \{L, R, A, B\}$
  - ▶  $\text{dom}(t_A) = \text{dom}(t_B) = \{L, R\}$
- ▶  $I = \{p \mapsto L, t_A \mapsto R, t_B \mapsto R\}$
- ▶  $O = \{\text{pickup}_{i,j} \mid i \in \{A, B\}, j \in \{L, R\}\} \cup \{\text{drop}_{i,j} \mid i \in \{A, B\}, j \in \{L, R\}\} \cup \{\text{move}_{i,j,j'} \mid i \in \{A, B\}, j, j' \in \{L, R\}, j \neq j'\}$ , where
  - ▶  $\text{pickup}_{i,j} = \langle t_i = j \wedge p = j, p := i, 1 \rangle$
  - ▶  $\text{drop}_{i,j} = \langle t_i = j \wedge p = i, p := j, 1 \rangle$
  - ▶  $\text{move}_{i,j,j'} = \langle t_i = j, t_i := j', 1 \rangle$
- ▶  $\gamma = (p = R)$

## Transition System of Example Task



- ▶ State  $\{p \mapsto i, t_A \mapsto j, t_B \mapsto k\}$  is depicted as  $ijk$ .
- ▶ Transition labels are again not shown. For example, the transition from LLL to ALL has the label  $\text{pickup}_{A,L}$ .

## D2.2 Abstractions

## Abstractions

### Definition (Abstraction)

Let  $\mathcal{T} = \langle S, L, c, T, s_0, S_* \rangle$  be a transition system.

An **abstraction** (also: **abstraction function**, **abstraction mapping**) of  $\mathcal{T}$  is a function  $\alpha : S \rightarrow S^\alpha$  defined on the states of  $\mathcal{T}$ , where  $S^\alpha$  is an arbitrary set.

Without loss of generality, we require that  $\alpha$  is surjective.

**Intuition:**  $\alpha$  maps the states of  $\mathcal{T}$  to another (usually smaller) **abstract** state space.

## Abstract Transition System

### Definition (Abstract Transition System)

Let  $\mathcal{T} = \langle S, L, c, T, s_0, S_* \rangle$  be a transition system, and let  $\alpha : S \rightarrow S^\alpha$  be an abstraction of  $\mathcal{T}$ .

The **abstract transition system induced by  $\alpha$** , in symbols  $\mathcal{T}^\alpha$ , is the transition system  $\mathcal{T}^\alpha = \langle S^\alpha, L, c, T^\alpha, s_0^\alpha, S_*^\alpha \rangle$  defined by:

- ▶  $T^\alpha = \{ \langle \alpha(s), \ell, \alpha(t) \rangle \mid \langle s, \ell, t \rangle \in T \}$
- ▶  $s_0^\alpha = \alpha(s_0)$
- ▶  $S_*^\alpha = \{ \alpha(s) \mid s \in S_* \}$

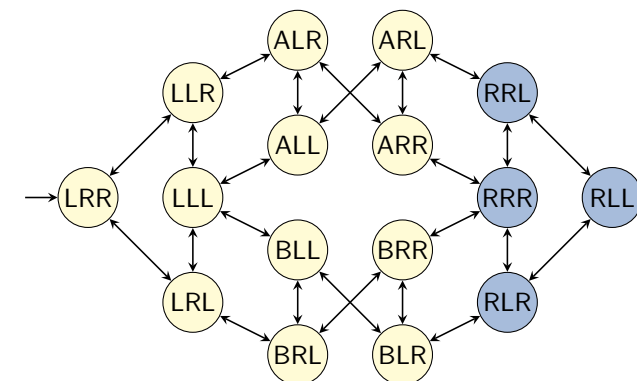
## Concrete and Abstract State Space

Let  $\mathcal{T}$  be a transition system and  $\alpha$  be an abstraction of  $\mathcal{T}$ .

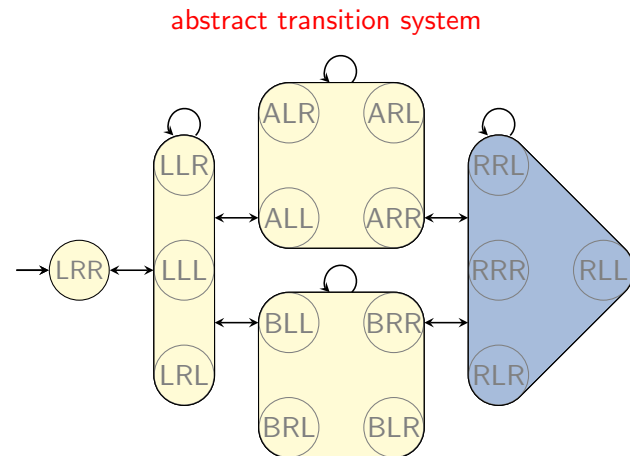
- ▶  $\mathcal{T}$  is called the **concrete transition system**.
- ▶  $\mathcal{T}^\alpha$  is called the **abstract transition system**.
- ▶ Similarly: **concrete/abstract state space**, **concrete/abstract transition**, etc.

## Abstraction: Example

concrete transition system



## Abstraction: Example



**Note:** Most arcs represent many parallel transitions.

## D2.3 Homomorphisms and Isomorphisms

## Homomorphisms and Isomorphisms

- ▶ The abstraction mapping  $\alpha$  that transforms  $\mathcal{T}$  to  $\mathcal{T}^\alpha$  is also called a **strict homomorphism** from  $\mathcal{T}$  to  $\mathcal{T}^\alpha$ .
- ▶ Roughly speaking, in mathematics a **homomorphism** is a property-preserving mapping between structures.
- ▶ A **strict** homomorphism is one where no additional features are introduced. A non-strict homomorphism in planning would mean that the abstract transition system may include additional transitions and goal states not induced by  $\alpha$ .
- ▶ We only consider strict homomorphisms in this course.
- ▶ If  $\alpha$  is bijective, it is called an **isomorphism** between  $\mathcal{T}$  and  $\mathcal{T}^\alpha$ , and the two transition systems are called **isomorphic**.

## Isomorphic Transition Systems

The notion of isomorphic transition systems is important enough to warrant a formal definition:

### Definition (Isomorphic Transition Systems)

Let  $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$  and  $\mathcal{T}' = \langle S', L', c', T', s'_0, S'_\star \rangle$  be transition systems.

We say that  $\mathcal{T}$  is **isomorphic to  $\mathcal{T}'$** , in symbols  $\mathcal{T} \sim \mathcal{T}'$ , if there exist bijective functions  $\varphi : S \rightarrow S'$  and  $\lambda : L \rightarrow L'$  such that:

- ▶  $s \xrightarrow{\ell} t \in T$  iff  $\varphi(s) \xrightarrow{\lambda(\ell)} \varphi(t) \in T'$ ,
- ▶  $c'(\lambda(\ell)) = c(\ell)$  for all  $\ell \in L$ ,
- ▶  $\varphi(s_0) = s'_0$ , and
- ▶  $s \in S_\star$  iff  $\varphi(s) \in S'_\star$ .

## Graph-Equivalent Transition Systems

Sometimes a weaker notion of equivalence is useful:

### Definition (Graph-Equivalent Transition Systems)

Let  $\mathcal{T} = \langle S, L, c, T, s_0, S_* \rangle$  and  $\mathcal{T}' = \langle S', L', c', T', s'_0, S'_* \rangle$  be transition systems.

We say that  $\mathcal{T}$  is **graph-equivalent** to  $\mathcal{T}'$ , in symbols  $\mathcal{T} \stackrel{\mathcal{G}}{\sim} \mathcal{T}'$ , if there exists a bijective function  $\varphi : S \rightarrow S'$  such that:

- ▶ There is a transition  $s \xrightarrow{\ell} t \in T$  with  $c(\ell) = k$  iff there is a transition  $\varphi(s) \xrightarrow{\ell'} \varphi(t) \in T'$  with  $c'(\ell') = k$ ,
- ▶  $\varphi(s_0) = s'_0$ , and
- ▶  $s \in S_*$  iff  $\varphi(s) \in S'_*$ .

**Note:** The labels of  $\mathcal{T}$  and  $\mathcal{T}'$  do not matter except that transitions **of the same cost** must be preserved.

## Isomorphism vs. Graph Equivalence

- ▶  $(\sim)$  and  $(\stackrel{\mathcal{G}}{\sim})$  are equivalence relations.
- ▶ Two isomorphic transition systems are interchangeable for all practical intents and purposes.
- ▶ Two graph-equivalent transition systems are interchangeable for most intents and purposes.
- ▶ In particular, their goal distances are identical.
- ▶ Isomorphism implies graph equivalence, but not vice versa.

## D2.4 Abstraction Heuristics

### Abstraction Heuristics

#### Definition (Abstraction Heuristic)

Let  $\alpha : S \rightarrow S^\alpha$  be an abstraction of a transition system  $\mathcal{T}$ .

The **abstraction heuristic induced by  $\alpha$** , written  $h^\alpha$ , is the heuristic function  $h^\alpha : S \rightarrow \mathbb{R}_0^+ \cup \{\infty\}$  defined as

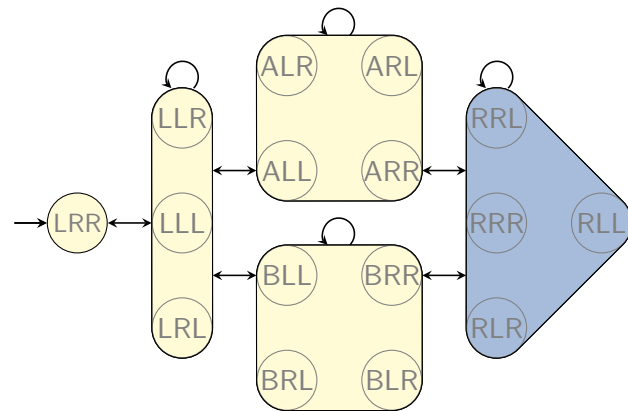
$$h^\alpha(s) = h_{\mathcal{T}^\alpha}^*(\alpha(s)) \quad \text{for all } s \in S,$$

where  $h_{\mathcal{T}^\alpha}^*$  denotes the goal distance function in  $\mathcal{T}^\alpha$ .

**Notes:**

- ▶  $h^\alpha(s) = \infty$  if no goal state of  $\mathcal{T}^\alpha$  is reachable from  $\alpha(s)$
- ▶ We also apply abstraction terminology to planning tasks  $\Pi$ , which stand for their induced transition systems. For example, an **abstraction of  $\Pi$**  is an abstraction of  $\mathcal{T}(\Pi)$ .

## Abstraction Heuristics: Example



$$h^\alpha(\{p \mapsto L, t_A \mapsto R, t_B \mapsto R\}) = 3$$

## Consistency of Abstraction Heuristics (1)

Theorem (Consistency and Admissibility of  $h^\alpha$ )

Let  $\alpha$  be an abstraction of a transition system  $\mathcal{T}$ .

Then  $h^\alpha$  is safe, goal-aware, admissible and consistent.

## Proof.

We prove goal-awareness and consistency; the other properties follow from these two.

Let  $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$ .

Let  $\mathcal{T}^\alpha = \langle S^\alpha, L, c, T^\alpha, s_0^\alpha, S_\star^\alpha \rangle$ .

**Goal-awareness:** We need to show that  $h^\alpha(s) = 0$  for all  $s \in S_\star$ , so let  $s \in S_\star$ . Then  $\alpha(s) \in S_\star^\alpha$  by the definition of abstract transition systems, and hence  $h^\alpha(s) = h_{\mathcal{T}^\alpha}^*(\alpha(s)) = 0$ . ...

## Consistency of Abstraction Heuristics (2)

## Proof (continued).

**Consistency:** Consider any state transition  $s \xrightarrow{\ell} t$  of  $\mathcal{T}$ .

We need to show  $h^\alpha(s) \leq c(\ell) + h^\alpha(t)$ .

By the definition of  $\mathcal{T}^\alpha$ , we get  $\alpha(s) \xrightarrow{\ell} \alpha(t) \in \mathcal{T}^\alpha$ .

Hence,  $\alpha(t)$  is a successor of  $\alpha(s)$  in  $\mathcal{T}^\alpha$  via the label  $\ell$ .

We get:

$$\begin{aligned} h^\alpha(s) &= h_{\mathcal{T}^\alpha}^*(\alpha(s)) \\ &\leq c(\ell) + h_{\mathcal{T}^\alpha}^*(\alpha(t)) \\ &= c(\ell) + h^\alpha(t), \end{aligned}$$

where the inequality holds because perfect goal distances  $h_{\mathcal{T}^\alpha}^*$  are consistent in  $\mathcal{T}^\alpha$ .

(The shortest path from  $\alpha(s)$  to the goal in  $\mathcal{T}^\alpha$  cannot be longer than the shortest path from  $\alpha(s)$  to the goal via  $\alpha(t)$ .)  $\square$

## D2.5 Coarsenings and Refinements

## Abstractions of Abstractions

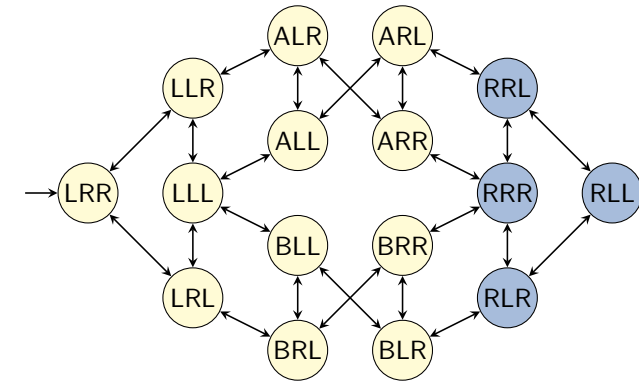
Since abstractions map transition systems to transition systems, they are **composable**:

- ▶ Using a first abstraction  $\alpha : S \rightarrow S'$ , map  $\mathcal{T}$  to  $\mathcal{T}^\alpha$ .
- ▶ Using a second abstraction  $\beta : S' \rightarrow S''$ , map  $\mathcal{T}^\alpha$  to  $(\mathcal{T}^\alpha)^\beta$ .

The result is **the same** as directly using the abstraction  $(\beta \circ \alpha)$ :

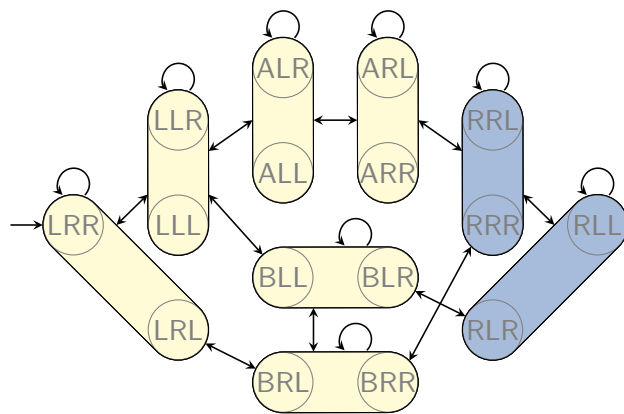
- ▶ Let  $\gamma : S \rightarrow S''$  be defined as  $\gamma(s) = (\beta \circ \alpha)(s) = \beta(\alpha(s))$ .
- ▶ Then  $\mathcal{T}^\gamma = (\mathcal{T}^\alpha)^\beta$ .

## Abstractions of Abstractions: Example (1)



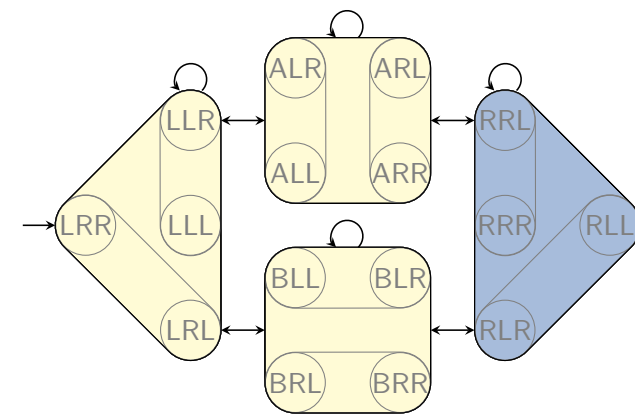
transition system  $\mathcal{T}$

## Abstractions of Abstractions: Example (2)



Transition system  $\mathcal{T}'$  as an abstraction of  $\mathcal{T}$

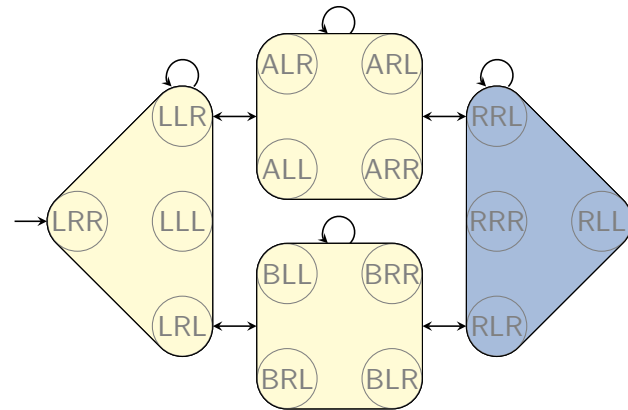
## Abstractions of Abstractions: Example (3)



Transition system  $\mathcal{T}''$  as an abstraction of  $\mathcal{T}'$



## Abstractions of Abstractions: Example (3)



Transition system  $\mathcal{T}''$  as an abstraction of  $\mathcal{T}$

## Coarsenings and Refinements

### Definition (Coarsening and Refinement)

Let  $\alpha$  and  $\gamma$  be abstractions of the same transition system such that  $\gamma = \beta \circ \alpha$  for some function  $\beta$ .

Then  $\gamma$  is called a **coarsening** of  $\alpha$  and  $\alpha$  is called a **refinement** of  $\gamma$ .

## Heuristic Quality of Refinements

### Theorem (Heuristic Quality of Refinements)

Let  $\alpha$  and  $\gamma$  be abstractions of the same transition system such that  $\alpha$  is a refinement of  $\gamma$ .

Then  $h^\alpha$  dominates  $h^\gamma$ .

In other words,  $h^\gamma(s) \leq h^\alpha(s) \leq h^*(s)$  for all states  $s$ .

## Heuristic Quality of Refinements: Proof

### Proof.

Since  $\alpha$  is a refinement of  $\gamma$ , there exists a function  $\beta$  with  $\gamma = \beta \circ \alpha$ .

For all states  $s$  of  $\Pi$ , we get:

$$\begin{aligned} h^\gamma(s) &= h_{\mathcal{T}^\gamma}^*(\gamma(s)) \\ &= h_{\mathcal{T}^\gamma}^*(\beta(\alpha(s))) \\ &= h_{\mathcal{T}^\alpha}^\beta(\alpha(s)) \\ &\leq h_{\mathcal{T}^\alpha}^*(\alpha(s)) \\ &= h^\alpha(s), \end{aligned}$$

where the inequality holds because  $h_{\mathcal{T}^\alpha}^\beta$  is an admissible heuristic in the transition system  $\mathcal{T}^\alpha$ .  $\square$

## D2.6 Summary

## Summary

- ▶ An **abstraction** is a function  $\alpha$  that maps the states  $S$  of a transition system to another (usually smaller) set  $S^\alpha$ .
- ▶ This **induces** an abstract transition system  $\mathcal{T}^\alpha$ , which behaves like the original transition system  $\mathcal{T}$  except that states mapped to the same abstract state cannot be distinguished.
- ▶ Abstractions  $\alpha$  induce **abstraction heuristics**  $h^\alpha$ :  $h^\alpha(s)$  is the goal distance of  $\alpha(s)$  in the abstract transition system.
- ▶ Abstraction heuristics are **safe**, **goal-aware**, **admissible** and **consistent**.
- ▶ Abstractions can be composed, leading to **coarser** vs. **finer** abstractions. Heuristics for finer abstractions dominate those for coarser ones.