# Latest Trends in Abstraction Heuristics for Classical Planning

## 1. Planning and Abstractions

Malte Helmert  Jendrik Seipp  Silvan Sievers

ICAPS 2015 Tutorial

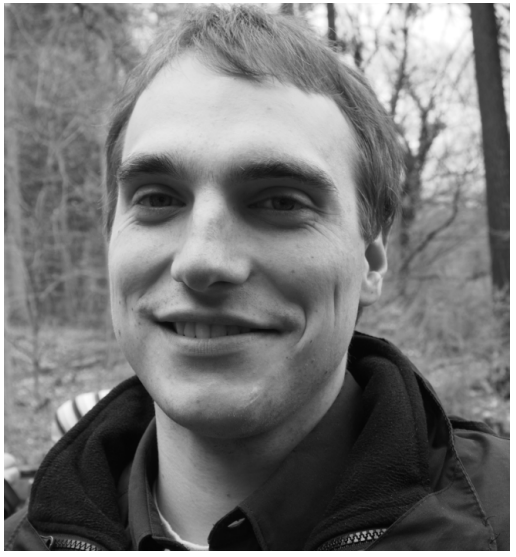
June 7, 2015

# About This Tutorial

# About Us

Malte

Jendrik



Silvan

Questions? Don't be shy to talk to us and/or email!

- malte.helmert@unibas.ch
- jendrik.seipp@unibas.ch
- silvan.sievers@unibas.ch

# Target Audience

## Target Audience

Ideally:

- You know what classical planning is.
  keywords: STRIPS, $SAS^+$
- You know what planning as heuristic search is.
  keywords: $A^*$, admissible heuristic, consistent heuristic
- You have a basic familiarity with abstraction heuristics and want to find out more about using them for planning.
  keywords: pattern databases, PDB heuristics

Please ask questions at any time!

# Tutorial Structure

1. Planning and Abstractions
2. Cartesian Abstractions
3. Merge-and-Shrink Abstraction
4. Outlook

# Planning

# Planning Tasks

## Definition (SAS$^+$ planning task)

A SAS$^+$ planning task is a 4-tuple $\Pi = \langle V, O, s_0, s_\star \rangle$:

- $V$: finite set of state variables,
  each variable $v \in V$ with finite domain $dom(v)$
- $O$: finite set of operators (actions) $o$ with
  - preconditions $pre(o)$ (partial variable assignment)
  - effects $eff(o)$ (partial variable assignment)
  - cost $cost(o)$ (number in $\mathbb{R}_{\geq 0}$)

  We write operators as $\langle pre(o), eff(o), cost(o) \rangle$
  and may omit $cost(o)$ if it is 1.
- $s_0$: initial state (variable assignment)
- $s_\star$: goal description (partial variable assignment)

# Planning

Informal definition of the planning problem:

## Classical Planning

Given: $SAS^+$ planning task

Find: plan (action sequence) leading from the initial state to a goal state (or show that no plan exists)

Additional soft or hard constraint:
minimize cost of plan (sum of costs of included actions)

$\rightsquigarrow$ full formal semantics via transition systems

# Transition Systems

## Definition (transition system)

A transition system is a 5-tuple $\Theta = \langle S, L, T, s_0, S_\star \rangle$:

- $S$: finite set of states
- $L$: finite set of transition labels,
  each label $\ell$ with associated cost $cost(\ell)$
- $T \subseteq S \times L \times S$: labelled transitions
- $s_0 \in S$: initial state
- $S_\star \subseteq S$: goal states

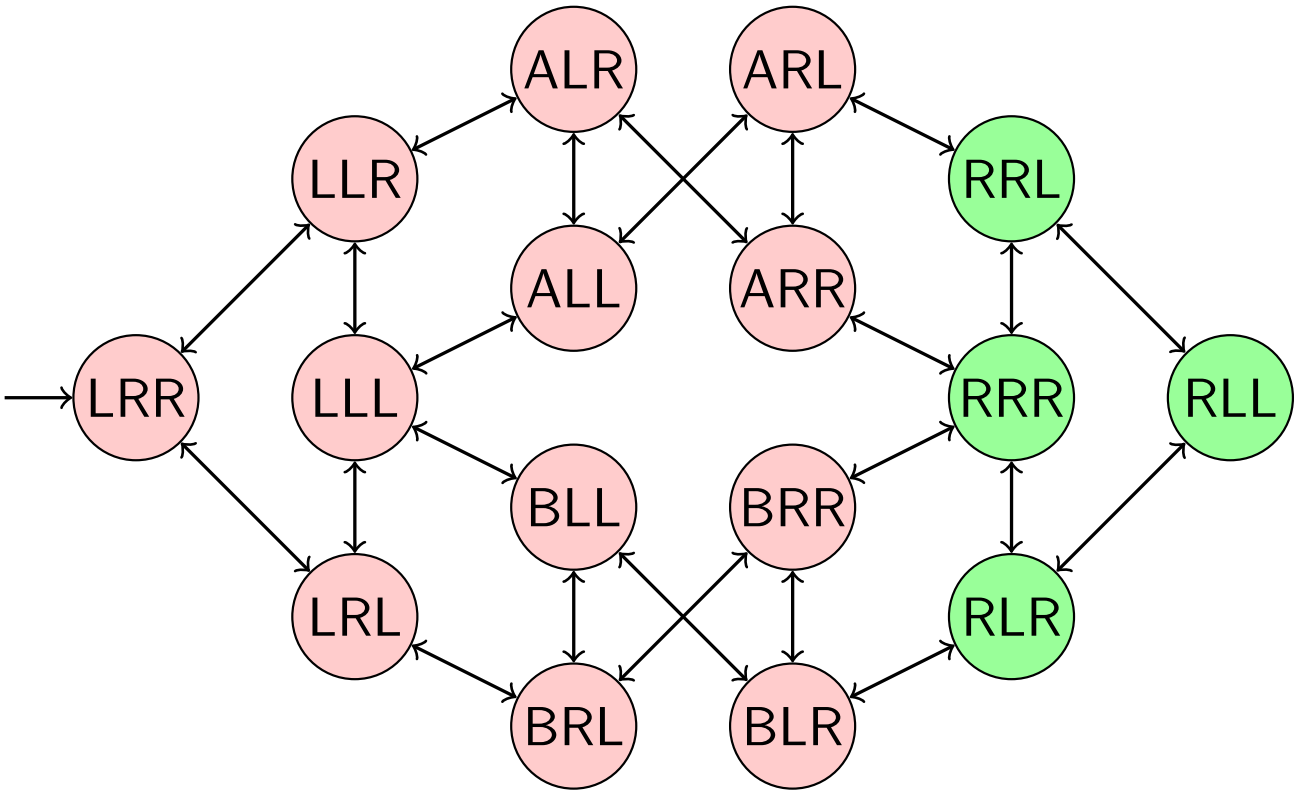
$\rightsquigarrow$ also called state spaces

# Transition Systems Induced by Planning Tasks

A $SAS^+$ task $\Pi$ induces a transition system $\Theta(\Pi)$:

- states: states of $\Pi$
- transition labels: operators of $\Pi$ (same cost function)
- transitions: transition $\langle s, o, t \rangle$ present iff:
  - $pre(o) \subseteq s$,
  - $eff(o) \subseteq t$ and
  - $s$ and $t$ agree on all variables not appearing in $pre(o)$ or $eff(o)$
- initial state: initial state of $\Pi$
- goal state: all states $s$ which agree with the goal of $\Pi$

# Induced Transition System Example




(transition labels omitted)

- one package, two trucks, two locations
- state variable package: $\{L, R, A, B\}$
- state variable truck A: $\{L, R\}$
- state variable truck B: $\{L, R\}$

# Abstractions

# Planning with Abstraction Heuristics

## Optimal Planning as Heuristic Search

Common approach for planning:
A$^*$ algorithm + admissible heuristic

## Admissible Heuristics

heuristic: $h : S \to \mathbb{R}_{\geq 0} \cup \{\infty\}$

- map states to cost-to-goal estimates
- admissible: do not overestimate goal distance

## Abstraction Heuristics

heuristic estimate is cost-to-goal in abstract transition system
(smaller state space) obtained as abstraction of real state space

# Abstractions: Formally

**Definition (abstraction)**

An abstraction of a transition system $\Theta$ with states $S$ is a function $\alpha : S \to S'$.

- $\alpha(s)$: abstract state for (concrete) state $s$
- idea: drop distinction between states $s_1$ and $s_2$ if mapped to same abstract state ($\alpha(s_1) = \alpha(s_2)$)
- alternative view: equivalence relation over states: ($s_1 \sim_\alpha s_2$ iff $\alpha(s_1) = \alpha(s_2)$)

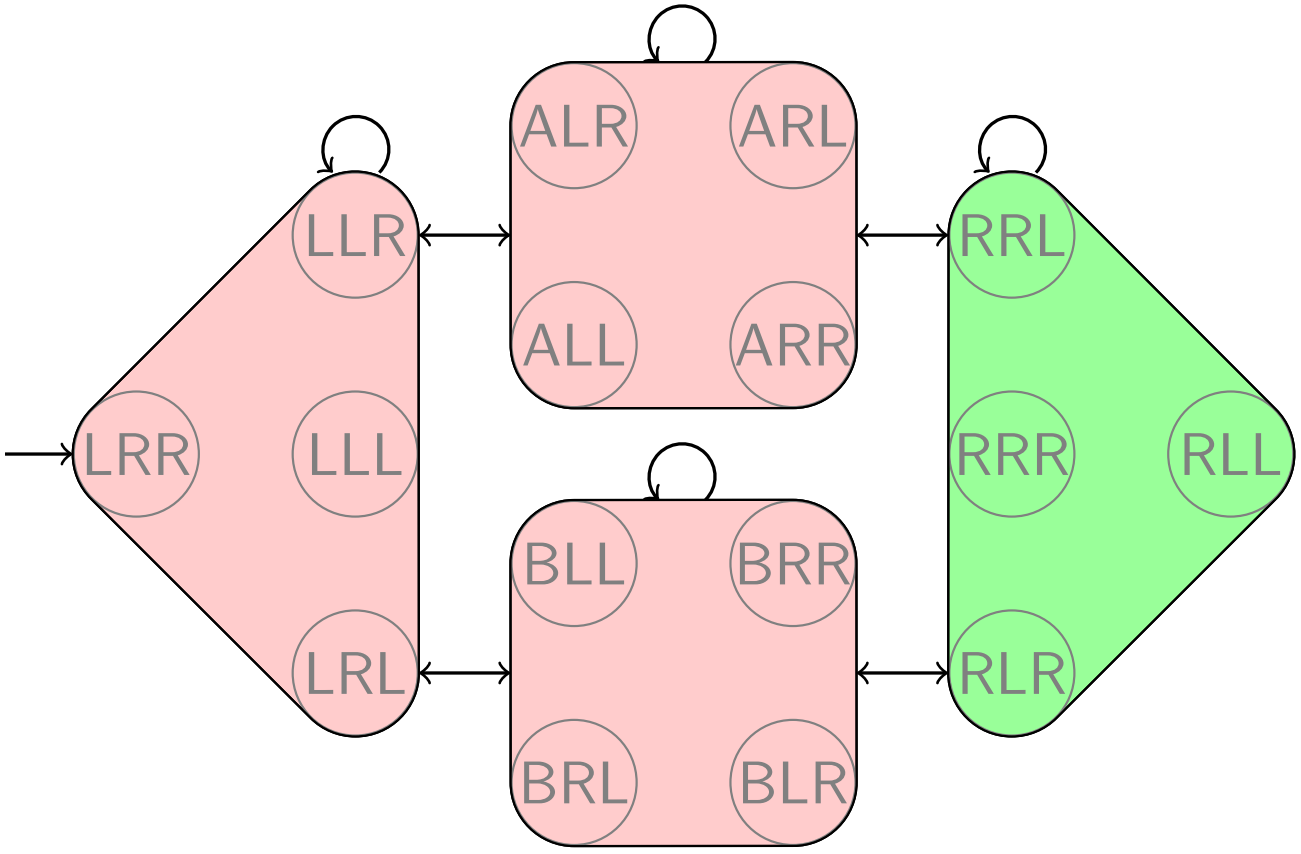
# Abstract Transition System

Abstraction $\alpha$ of concrete transition system $\Theta = \langle S, L, T, s_0, S_\star \rangle$ induces abstract transition system $\alpha(\Theta)$:

- states: $\{\alpha(s) \mid s \in S\}$
- transition labels: $L$ (same as concrete)
- transitions: transition $\langle \alpha(s), \ell, \alpha(t) \rangle$ induced by every concrete transition $\langle s, \ell, t \rangle \in T$
- initial state: $\alpha(s_0)$
- goal state: goal state $\alpha(s_\star)$ induced by every concrete goal state $s_\star \in S_\star$

abstraction heuristic: $h^\alpha(s) =$ cost-to-goal from $\alpha(s)$ in $\alpha(\Theta)$.

# Abstract Transition System: Example




$$\rightsquigarrow h^{\alpha}(s_0) = 2$$

# Classes of Abstractions

# Useful Abstractions

Conflicting goals in using abstractions for planning:

- obtain informative heuristic
- keep representation small

Abstractions have small representations if they have

- few abstract states $^{(*)}$
- succinct encoding for $\alpha$

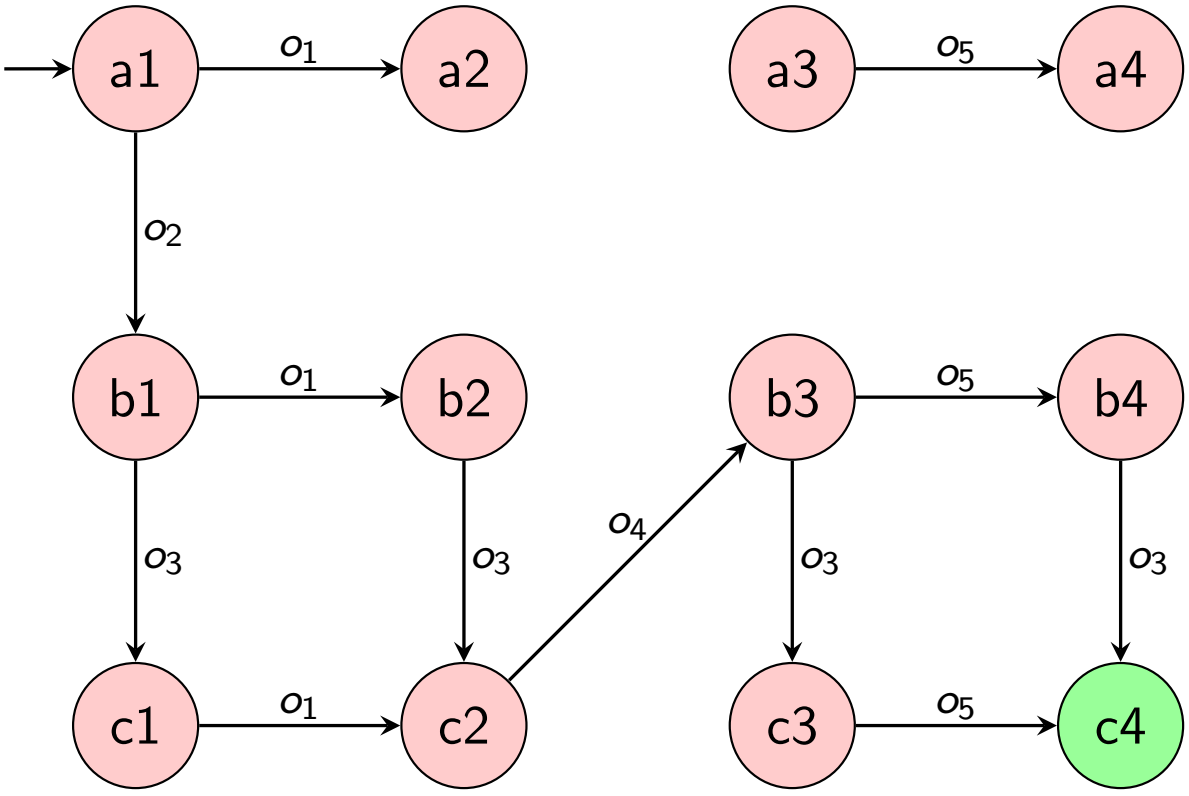
$^{(*)}$: but see also symbolic and implicit abstractions

# Four Classes of Abstractions

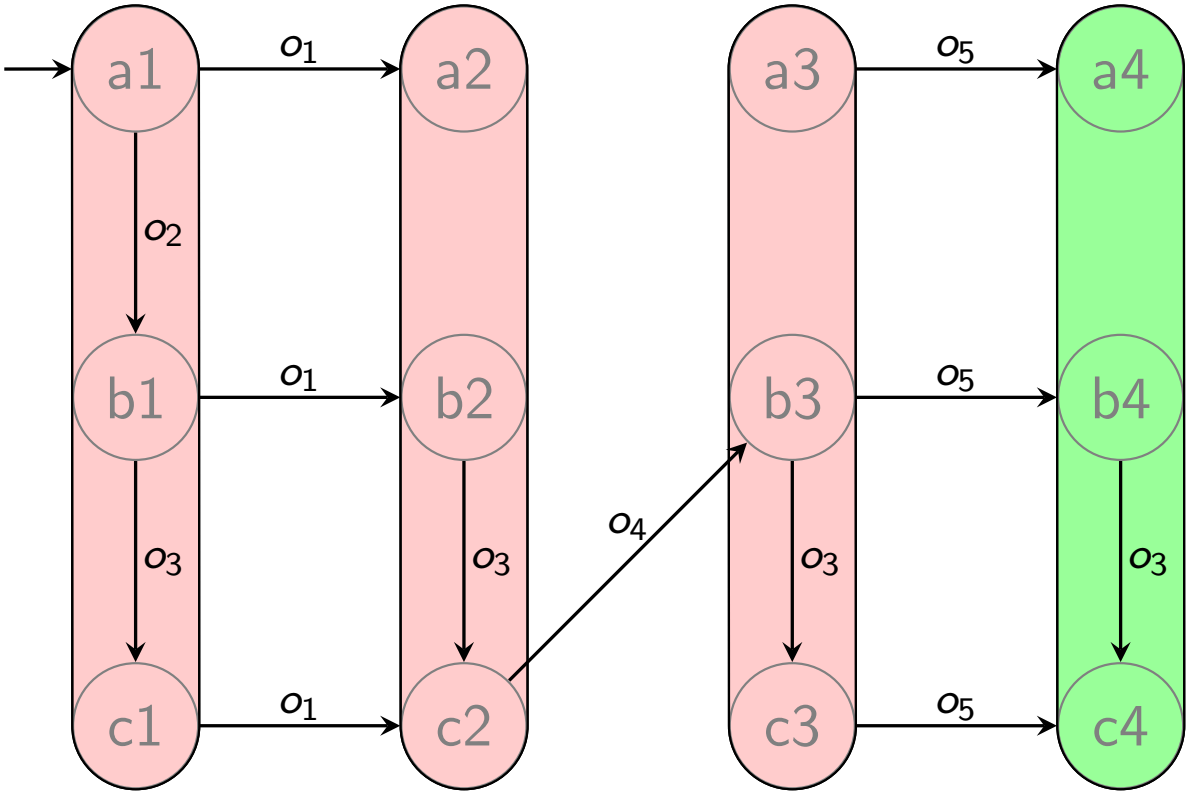
Some classes of abstractions studied in planning and heuristic search, in increasing order of generality:

1. projections ($\rightsquigarrow$ pattern databases)
2. domain abstractions
3. Cartesian abstractions
   $\rightsquigarrow$ part 2 of this tutorial
4. merge-and-shrink abstractions
   $\rightsquigarrow$ part 3 of this tutorial

# Concrete Transition System Example

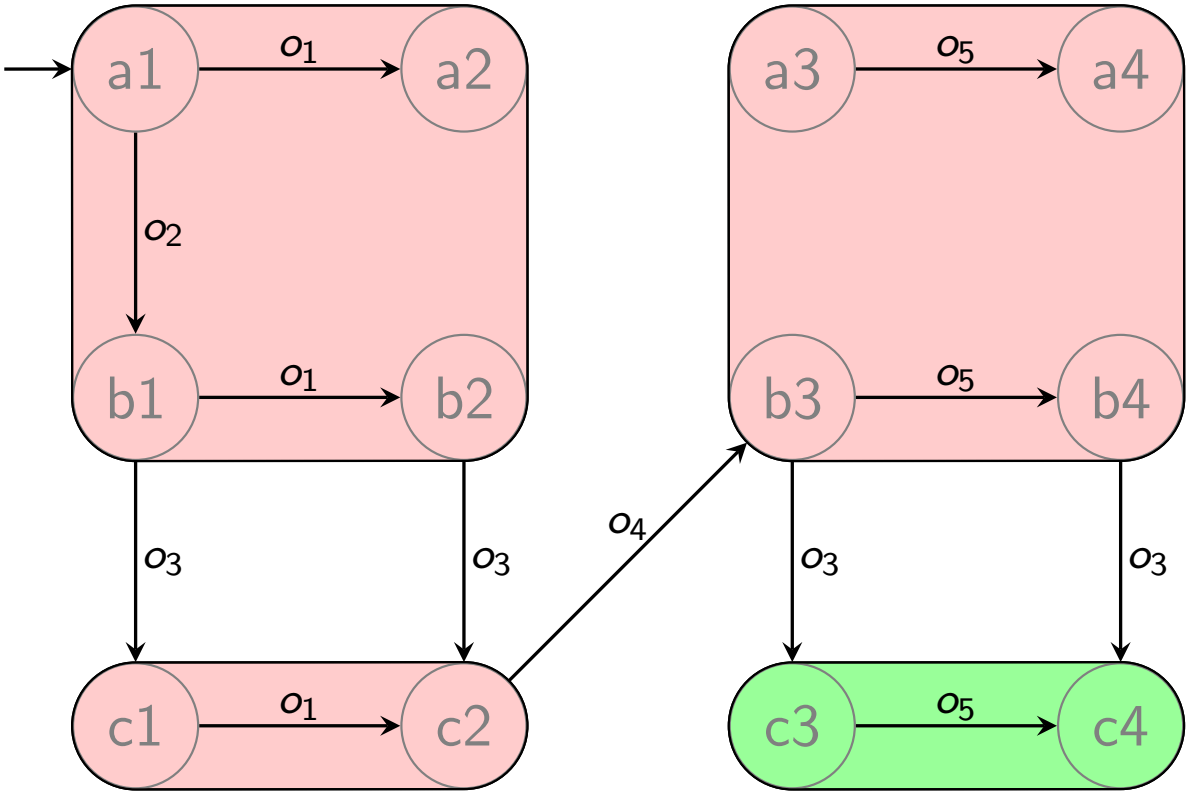


# Projection/Pattern Database Example



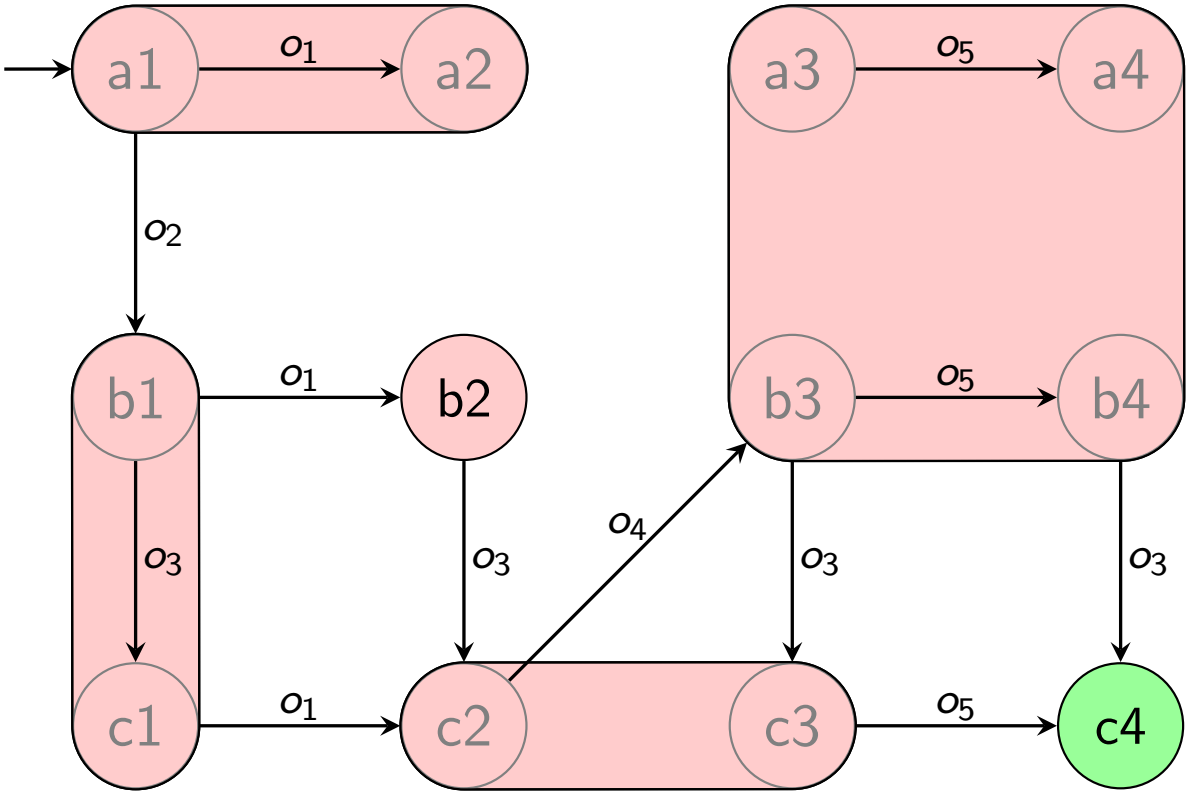

projection ($\rightsquigarrow$ pattern database)

# Domain Abstraction Example



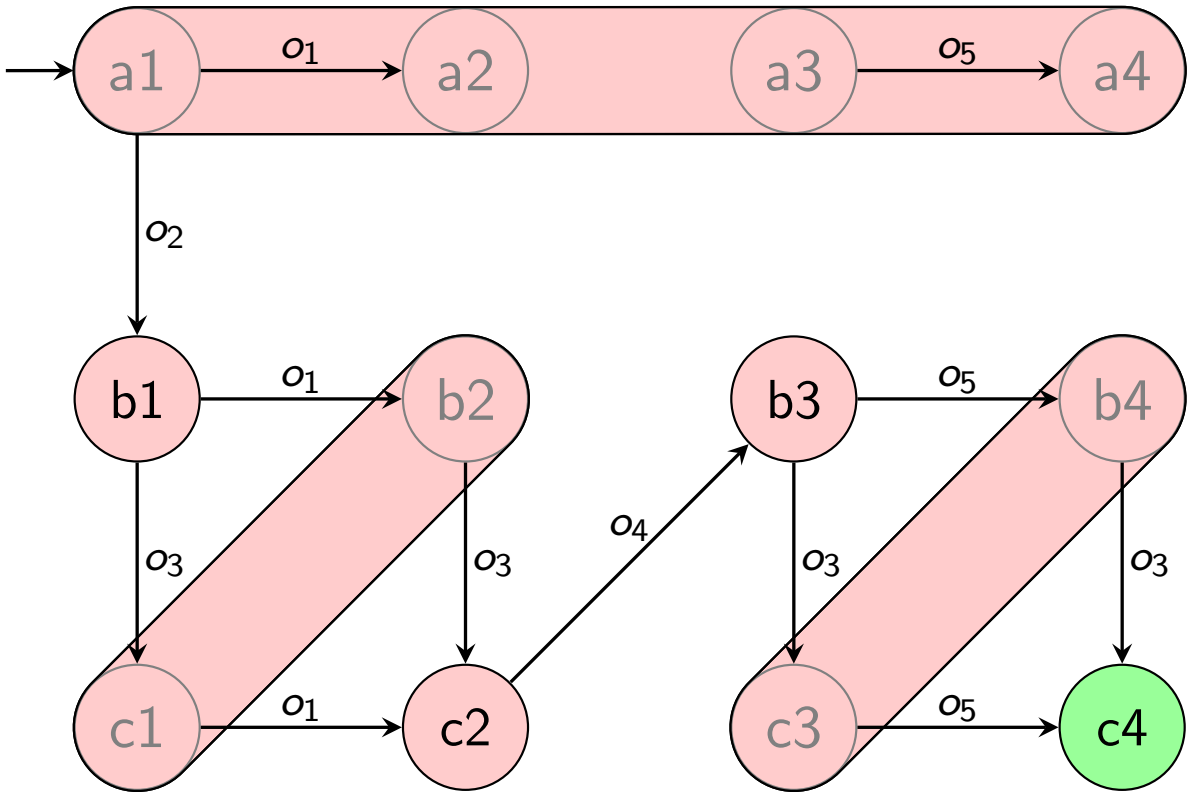

domain abstraction

# Cartesian Abstraction Example




Cartesian abstraction

# Merge-and-Shrink Abstraction Example




merge-and-shrink abstraction