

Planning and Optimization

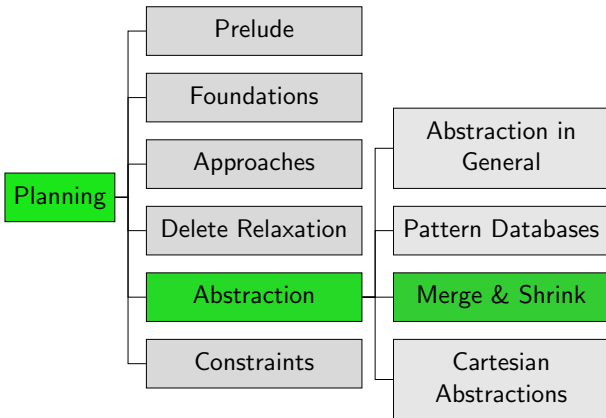
E9. Merge-and-Shrink: Factored Transition Systems

Malte Helmert and Gabriele Röger

Universität Basel

November 17, 2025

Content of the Course

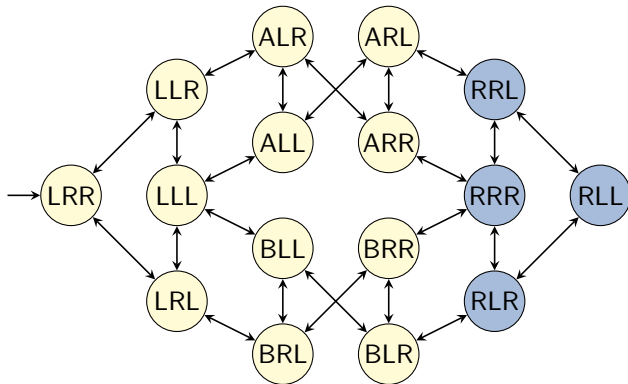


Motivation

Beyond Pattern Databases

- Despite their popularity, pattern databases have some **fundamental limitations** ($\rightsquigarrow$ example on next slides).
- Today and next time, we study a class of abstractions called **merge-and-shrink abstractions**.
- Merge-and-shrink abstractions can be seen as a **proper generalization** of pattern databases.
 - They can do everything that pattern databases can do (modulo polynomial extra effort).
 - They can do some things that pattern databases cannot.

Back to the Running Example

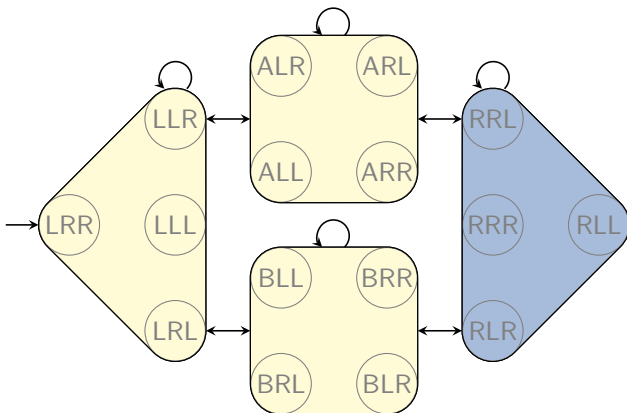


Logistics problem with one package, two trucks, two locations:

- state variable **package**: $\{L, R, A, B\}$
- state variable **truck A**: $\{L, R\}$
- state variable **truck B**: $\{L, R\}$

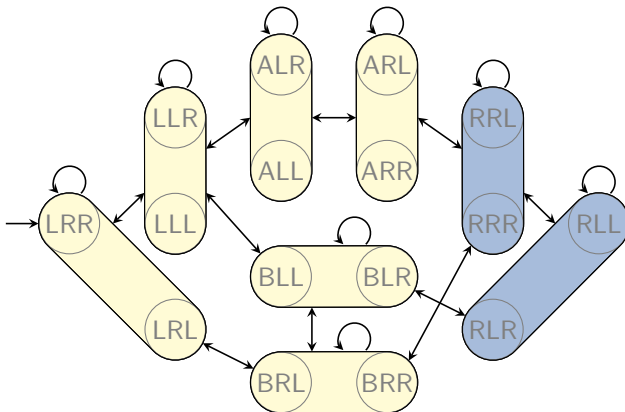
Example: Projection (1)

$\mathcal{T}^{\pi}\{\text{package}\}:$



Example: Projection (2)

$\mathcal{T}^\pi\{\text{package, truck A}\}:$



Limitations of Projections

How accurate is the PDB heuristic?

- consider **generalization of the example**:
 N trucks, 1 package
- consider **any** pattern that is a proper subset of variable set V
- $h(s_0) \leq 2 \rightsquigarrow$ **no better** than atomic projection to **package**

These values cannot be improved by maximizing over several patterns or using additive patterns.

Merge-and-shrink abstractions can represent heuristics with $h(s_0) \geq 3$ for tasks of this kind of any size.

Time and space requirements are **linear in N** .

(In fact, with time/space $O(N^2)$ we can construct a merge-and-shrink abstraction that gives the **perfect heuristic h^*** for such tasks, but we do not show this here.)

Main Idea

Merge-and-Shrink Abstractions: Main Idea

Main Idea of Merge-and-shrink Abstractions

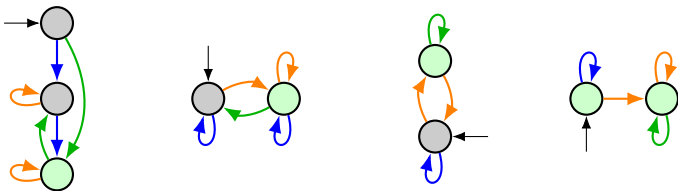
(due to Dräger, Finkbeiner & Podelski, 2006):

Instead of **perfectly** reflecting **a few** state variables,
reflect **all** state variables, but in a **potentially lossy** way.

- Represent planning task as **factored transition system** (FTS):
a set of (small) abstract transition systems (**factors**)
that jointly represent the full transition system of the task.
- Iteratively **transform** FTS by:
 - **merging**: combining two factors into one
 - **shrinking**: reducing the size of a single factor by abstraction
- When only a single factor is left, its goal distances
are the merge-and-shrink heuristic values.

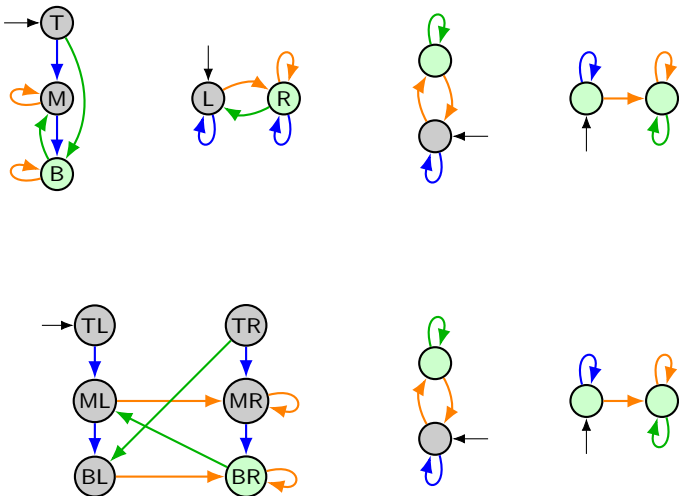
Merge-and-Shrink Abstractions: Idea

Start from atomic factors (projections to single state variables)



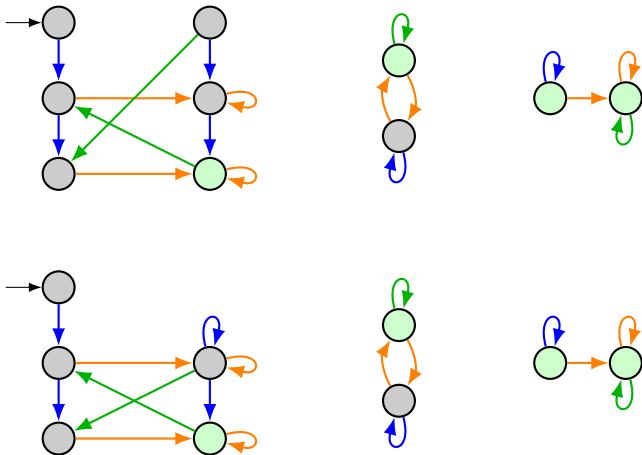
Merge-and-Shrink Abstractions: Idea

Merge: replace two factors with their product



Merge-and-Shrink Abstractions: Idea

Shrink: replace a factor by an abstraction of it



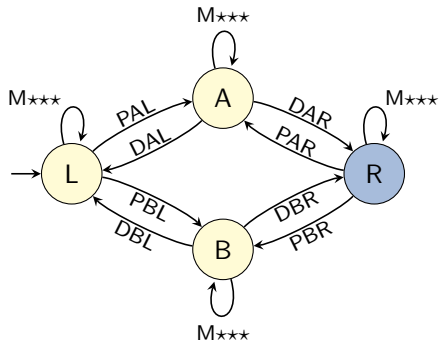
Atomic Projections

Running Example: Explanations

- **Atomic projections** (projections to a single state variable) play an important role for merge-and-shrink abstractions.
- Unlike previous chapters, **transition labels** are critically important for merge-and-shrink.
- Hence we now look at the transition systems for atomic projections of our example task, including transition labels.
- We abbreviate labels (operator names) as in these examples:
 - **MALR**: **m**ove truck **A** from **l**eft to **r**ight
 - **DAR**: **d**rop package from truck **A** at **r**ight location
 - **PBL**: **p**ick up package with truck **B** at **l**eft location
- We abbreviate parallel arcs with **commas** and **wildcards** (★) as in these examples:
 - **PAL, DAL**: two parallel arcs labeled **PAL** and **DAL**
 - **MA★**: two parallel arcs labeled **MALR** and **MARL**

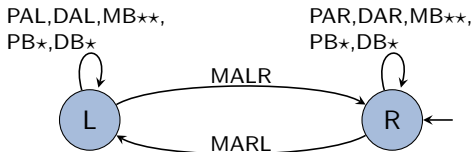
Running Example: Atomic Projection for Package

$\mathcal{T}^{\pi_{\{\text{package}\}}}$:



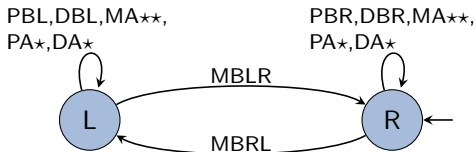
Running Example: Atomic Projection for Truck A

$\mathcal{T}^{\pi\{\text{truck A}\}}:$



Running Example: Atomic Projection for Truck B

$\mathcal{T}^{\pi\{\text{truck B}\}}:$



Synchronized Product

Synchronized Product: Idea

- Given two abstract transition systems with the same labels, we can compute a **product transition system**.
- The product transition system **captures all information** of both transition systems.
- A sequence of labels is a solution for the product iff it is a solution for both factors.

Synchronized Product of Transition Systems

Definition (Synchronized Product of Transition Systems)

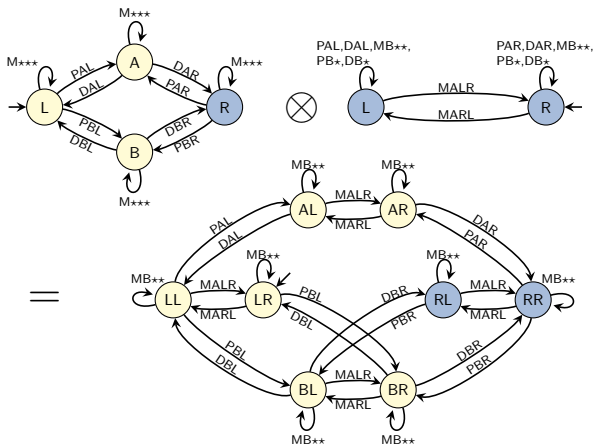
For $i \in \{1, 2\}$, let $\mathcal{T}_i = \langle S_i, L, c, T_i, s_{0i}, S_{\star i} \rangle$ be transition systems with the same labels and cost function.

The **synchronized product** of $\mathcal{T}_1$ and $\mathcal{T}_2$, in symbols $\mathcal{T}_1 \otimes \mathcal{T}_2$, is the transition system $\mathcal{T}_{\otimes} = \langle S_{\otimes}, L, c, T_{\otimes}, s_{0\otimes}, S_{\star\otimes} \rangle$ with

- $S_{\otimes} = S_1 \times S_2$
- $T_{\otimes} = \{ \langle s_1, s_2 \rangle \xrightarrow{\ell} \langle t_1, t_2 \rangle \mid s_1 \xrightarrow{\ell} t_1 \in T_1 \text{ and } s_2 \xrightarrow{\ell} t_2 \in T_2 \}$
- $s_{0\otimes} = \langle s_{01}, s_{02} \rangle$
- $S_{\star\otimes} = S_{\star 1} \times S_{\star 2}$

Example: Synchronized Product

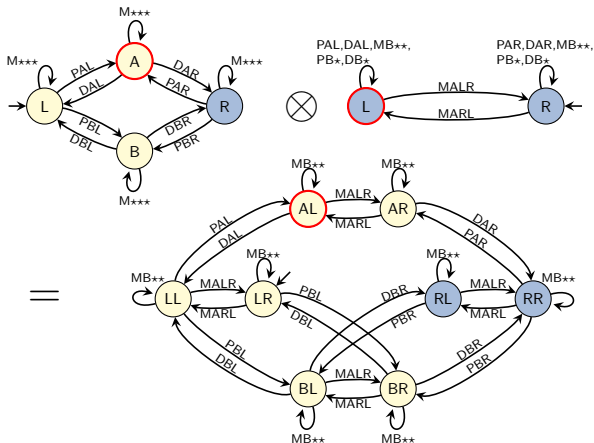
$$\mathcal{T}^{\pi}\{\text{package}\} \otimes \mathcal{T}^{\pi}\{\text{truck A}\}:$$



Example: Synchronized Product

$$\mathcal{T}^{\pi}\{\text{package}\} \otimes \mathcal{T}^{\pi}\{\text{truck A}\}:$$

$$S_{\otimes} = S_1 \times S_2$$



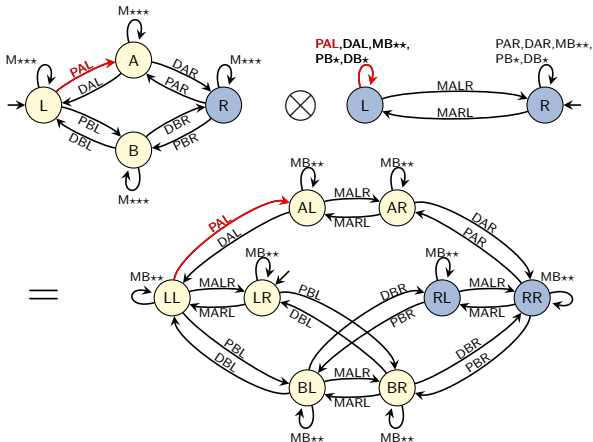
$$S_{\star \otimes} = S_{\star 1} \times S_{\star 2}$$

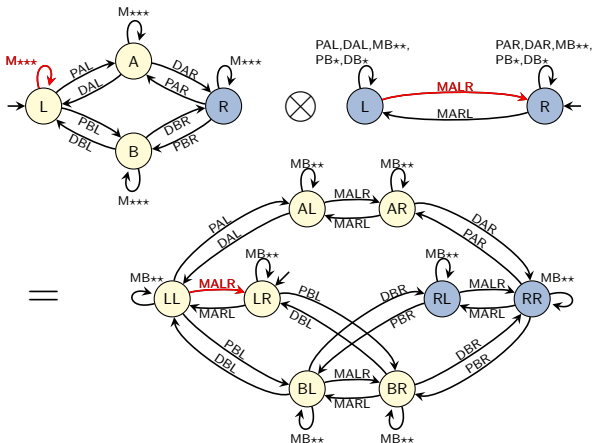


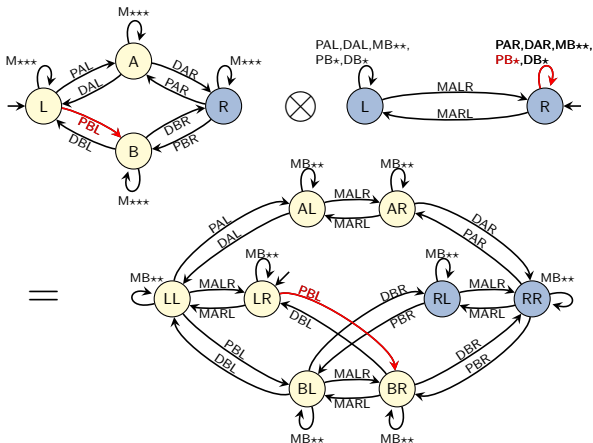
Example: Synchronized Product

$$\mathcal{T}^{\pi}\{\text{package}\} \otimes \mathcal{T}^{\pi}\{\text{truck A}\}:$$

$$T_{\otimes} = \{\langle s_1, s_2 \rangle \xrightarrow{\ell} \langle t_1, t_2 \rangle \mid s_1 \xrightarrow{\ell} t_1 \in T_1 \text{ and } s_2 \xrightarrow{\ell} t_2 \in T_2\}$$



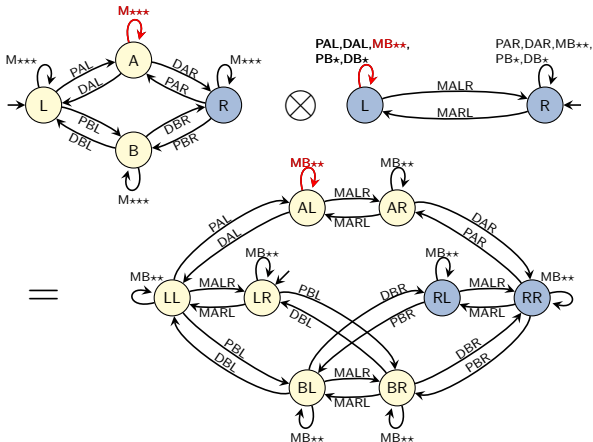




Example: Synchronized Product

$\mathcal{T}^{\pi}\{\text{package}\} \otimes \mathcal{T}^{\pi}\{\text{truck A}\}:$

$$T_{\otimes} = \{\langle s_1, s_2 \rangle \xrightarrow{\ell} \langle t_1, t_2 \rangle \mid s_1 \xrightarrow{\ell} t_1 \in T_1 \text{ and } s_2 \xrightarrow{\ell} t_2 \in T_2\}$$



Associativity and Commutativity

- Up to isomorphism (“names of states”), products are associative and commutative:
 - $(\mathcal{T} \otimes \mathcal{T}') \otimes \mathcal{T}'' \sim \mathcal{T} \otimes (\mathcal{T}' \otimes \mathcal{T}'')$
 - $\mathcal{T} \otimes \mathcal{T}' \sim \mathcal{T}' \otimes \mathcal{T}$
- We do not care about names of states and thus treat products as associative and commutative.
- We can then define the product of a **set** $F = \{\mathcal{T}_1, \dots, \mathcal{T}_n\}$ of transition systems: $\bigotimes F := \mathcal{T}_1 \otimes \dots \otimes \mathcal{T}_n$

Factored Transition Systems

Factored Transition System

Definition (Factored Transition System)

A finite set $F = \{\mathcal{T}_1, \dots, \mathcal{T}_n\}$ of transition systems with the same labels and cost function is called a **factored transition system (FTS)**.

F **represents** the transition system $\bigotimes F$.

A planning task gives rise to an FTS via its atomic projections:

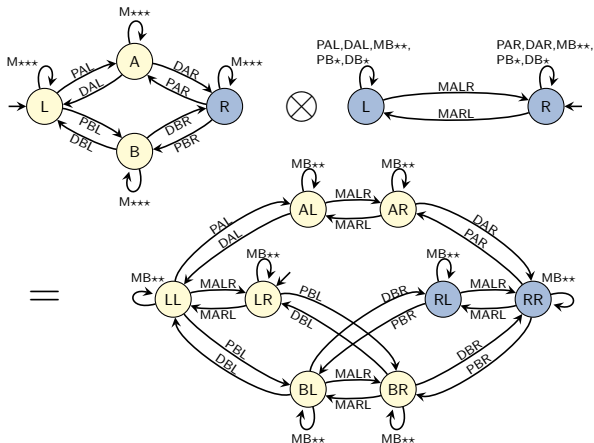
Definition (Factored Transition System Induced by Planning Task)

Let Π be a planning task with state variables V .

The **factored transition system induced by Π** is the FTS $F(\Pi) = \{\mathcal{T}^{\pi\{v\}} \mid v \in V\}$.

Back to the Example Product

$$\mathcal{T}^{\pi}\{\text{package}\} \otimes \mathcal{T}^{\pi}\{\text{truck A}\}:$$



We have $\mathcal{T}^{\pi}\{\text{package}\} \otimes \mathcal{T}^{\pi}\{\text{truck A}\} \sim \mathcal{T}^{\pi}\{\text{package, truck A}\}$. Coincidence?

Products of Projections

Theorem (Products of Projections)

Let Π be a **SAS⁺** planning task with variable set V ,
and let V_1 and V_2 be **disjoint subsets** of V .

Then $\mathcal{T}^{\pi_{V_1}} \otimes \mathcal{T}^{\pi_{V_2}} \sim \mathcal{T}^{\pi_{V_1 \cup V_2}}$.

(Proof omitted.)

$\rightsquigarrow$ products allow us to build finer projections from coarser ones

Recovering $\mathcal{T}(\Pi)$ from the Factored Transition System

- By repeated application of the theorem, we can recover **all pattern database heuristics** of a SAS^+ planning task as products of atomic factors.
- Moreover, by computing the product of **all** atomic projections, we can recover the **identity abstraction** $\text{id} = \pi_V$.

This implies:

Corollary (Recovering $\mathcal{T}(\Pi)$ from the Factored Transition System)

Let Π be a **SAS^+** planning task. Then $\bigotimes F(\Pi) \sim \mathcal{T}(\Pi)$.

This is an important result because it shows that $F(\Pi)$ **represents all important information** about Π .

Summary

Summary

- A **factored transition system** is a set of transition systems that represents a larger transition system by focusing on its individual components (**factors**).
- For planning tasks, these factors are the **atomic projections** (projections to single state variables).
- The **synchronized product** $\mathcal{T} \otimes \mathcal{T}'$ of two transition systems with the same labels captures their “joint behaviour”.
- For SAS^+ tasks, all **projections** can be obtained as products of atomic projections.
- In particular, the product of all factors of a SAS^+ task results in the **full** transition system of the task.