A Theory of Merge-and-Shrink for Stochastic Shortest Path Problems

Thorsten Klößner¹, Álvaro Torralba², Marcel Steinmetz¹, Silvan Sievers³

¹Saarland University, Germany
²Aalborg University, Denmark
³Basel University, Switzerland

{kloessner, steinmetz}@cs.uni-saarland.de, alto@cs.aau.dk, silvan.sievers@unibas.ch

Abstract
The merge-and-shrink framework is a powerful tool to construct state space abstractions based on factored representations. One of its core applications in classical planning is the construction of admissible abstraction heuristics. In this paper, we develop a compositional theory of merge-and-shrink in the context of probabilistic planning, focusing on stochastic shortest path problems (SSPs). As the basis for this development, we contribute a novel factored state space model for SSPs. We show how general transformations, including abstractions, can be formulated on this model to derive admissible and/or perfect heuristics. To formalize the merge-and-shrink framework for SSPs, we transfer the fundamental merge-and-shrink transformations from the classical setting: shrinking, merging, and label reduction. We analyze the formal properties of these transformations in detail and show how the conditions under which shrinking and label reduction lead to perfect heuristics can be extended to the SSP setting.

Merge-and-shrink (Dräger, Finkbeiner, and Podelski 2009) is a powerful framework based on factored state space representations with a broad range of applications in classical planning, including the computation of abstraction heuristics (e.g., Helmert et al. 2014; Sievers and Helmert 2021), task reformulation (Torralba and Sievers 2019) or identifying irrelevant actions (Torralba and Kissmann 2015). The core concept of the merge-and-shrink framework is to represent a (large) labelled transition system implicitly as a factored transition system (FTS), a set of multiple smaller transition systems (called factors) with the same labels. Given such an FTS of an input transition system, the framework repeatedly applies transformations, while at the same time maintaining a factored state mapping (Helmert, Köger, and Sievers 2015) from the original transition system to the transformed FTS. All transformations can be categorized in terms of desirable properties that they have. For example, conservative transformations induce admissible heuristics and exact transformations induce perfect heuristics. Furthermore, transformations can be chained, and such composed transformations inherit the properties of the individual transformations. As all merge-and-shrink transformations are least conservative, each factor of the transformed FTS induces an abstraction of the original transition system, which empowers computing admissible abstraction heuristics.

Introduction
Probabilistic planning is a generalization of classical planning in which actions are stochastic, i.e., an action may result in several outcomes, each known to occur with a specific probability. Stochastic shortest path problems (SSPs, Bertsekas and Tsitsiklis 1991) are particular probabilistic planning problems in which the objective is the minimization of accumulated costs in expectation until a specific goal is reached. Heuristic search algorithms (Hansen and Zilberstein 2001; Bonet and Geffner 2003; Trevizan et al. 2017) can solve SSPs without constructing the full state space by utilizing an admissible heuristic, i.e., a function that underestimates the expected cost-to-goal of all states. Previous work on admissible heuristics include determinization-based approaches (e.g., Bonet and Geffner 2005; Kolobov, Mausam, and Weld 2010, 2012), which evaluate classical planning heuristics on the all-outcomes determinization of the problem; occupation measure heuristics (Trevizan, Thiébaux, and Haslum 2017), which can be seen as a generalization of operator-counting heuristics from classical planning (Pommerening et al. 2014); and pattern database heuristics for SSPs (Klößner and Hoffmann 2021).
tion (Larsen and Skou 1991) are not exact, and subsequently adapt them to yield exactness. Our theory allows to fully exploit the power of merge-and-shrink for SSPs, including the computation of admissible heuristics, thus providing a method to compute arbitrary abstraction heuristics for SSPs.

Background

In this section, we introduce our probabilistic planning setting and recall the merge-and-shrink framework.

Mathematical Notation

We write $D(X) := \{\delta : X \rightarrow [0,1] | \sum_{x \in X} \delta(x) = 1\}$ for the set of (discrete) probability distributions over $X$ and $\text{supp}(\delta) := \{x | \delta(x) > 0\}$ for the support of $\delta \in D(X)$. We use $\mathbb{R} := \mathbb{R} \cup \{-\infty, \infty\}$ for the extended reals, with the convention $0 \cdot \pm\infty = 0$. We read $\text{id}_X$ for the identity function on $X$, using id if $X$ is clear from the context.

A partial function $f$ from a set $X$ to $Y$ is a function from some domain $\text{dom}(f) \subseteq X \rightarrow Y$ and is denoted $f : X \rightarrow Y$. If $\text{dom}(f) = X$ then $f$ is total on $X$ and we write $f : X \rightarrow Y$. The image of $X' \subseteq X$ under $f$ is denoted $f(X') := \{f(x) | x \in X' \cap \text{dom}(f)\}$ and the pre-image of $Y' \subseteq Y$ under $f$ is denoted $f^{-1}(Y') := \{x \in X | f(x) \in Y'\}$.

The restriction of $f$ to $X \subseteq \text{dom}(f)$ is denoted $f|_X$. The composition of $f : X \rightarrow Y$ and $g : Y \rightarrow Z$ is denoted $g \circ f : \text{dom}(f) \cap f^{-1}(\text{dom}(g)) \rightarrow Z, x \mapsto g(f(x))$. The disjoint union of $f : X' \rightarrow Y'$ and $g : X' \rightarrow Y'$ is denoted $f \cup g : X' \cup X' \rightarrow Y \cup Y'$, $x \mapsto f(x)$ if $x \in X'$ and $g(x)$ otherwise.

Probabilistic Planning

Stochastic Shortest Path Problems A stochastic shortest path problem (SSP, Bertsekas and Tsitsiklis 1991) is a tuple $S = \langle S, L, c, P, G \rangle$, where $S$ is a set of states, $L$ is a set of labels, $c : L \rightarrow \mathbb{R}_{\geq 0}$ is a cost function, $P : S \times L \times S \rightarrow \mathbb{R}_{\geq 0}$ is a transition probability function with $\sum_{\ell \in L} P(s, \ell, t) \in [0,1]$ for all $(s, \ell) \in S \times L$, and $G \subseteq S$ is a set of goal states. The set of labels applicable in $s \in S$ is denoted $\text{dom}(s) := \{\ell \in L | \sum_{\ell' \in L} P(s, \ell, t) = 1\}$.

A policy for $S$ is a function $\pi : S \rightarrow L$ either selecting a label $\pi(s) \in \text{dom}(s)$ when $s$ is encountered during execution resulting in the successor $t$ with probability $P(s, \pi(s), t)$, or deciding to terminate if $s \notin \text{dom}(s)$. An s-proper policy for $s \in S$ terminates in a goal state with probability 1 when executed from $s$. We write $\Pi(s)$ for the set of s-proper policies.

The expected cost-to-goal function $J_G^\pi(s)$ for $\pi$ is defined as $J_G^\pi(s) := \mathbb{E}_{\pi,s}[\sum_{t \geq 0} c(\pi(S_t))]$, where the random variable $S_t$ models the state after $t$ decisions of $\pi$ when starting in $s$ and $T := \inf\{t | S_{t+1} \notin \text{dom}(s)|\}$ is the last execution step before termination. If $T = \infty$, the sum is a series.

The optimal expected cost-to-goal function $J_G^* := \min_{\pi \in \Pi(s)} J_G^\pi(s)$, where $\mathbb{E}$ signifies that dead ends are unavoidable from $s$. A policy $\pi$ is optimal for $s \in S$ iff $J_G^\pi(s) = J_G^*(s)$. For $s \in S$, $J_G^*(s)$ is characterize by a linear program. To this end, let the Bellman (1957) optimality operator $B_S : (S \rightarrow \mathbb{R}_{\geq 0}) \rightarrow (S \rightarrow \mathbb{R}_{\geq 0})$ be defined by $(B_S J)(s) := \min_{\ell \in \text{dom}(s)} \{c(\ell) + \sum_{t \in S} P(s, \ell, t) J(t)| t \}$ if $L(s) \neq 0$, else $(B_S J)(s) := J(s)$. It holds that $J_G^*(s) = \sup_{J : S \rightarrow \mathbb{R}_{\geq 0}} \{J(s) | J \leq B_S J \land \forall s \in G. J(s) = 0\}$.

SSP Heuristic Search Given an SSP $S = \langle S, L, c, P, G \rangle$, a heuristic $h : S \rightarrow \mathbb{R}_{\geq 0}$ provides estimates for $J_G^*(s)$ for all states $s \in S$. $h$ is admissible iff $h \leq J_G^*$, pessimistic iff $h \geq J_G^*$ and perfect iff $h = J_G^*$. Furthermore, $h$ is goal-aware iff $h(s) = 0$ for all $s \in G$, consistent iff $h \leq B_S h$ and safe iff $J_G^*(s) \in \mathbb{R}_{\geq 0}$ implies $h(s) \in \mathbb{R}_{\geq 0}$. A goal-aware and consistent heuristic is not necessarily admissible if $\infty$ is allowed as a heuristic value, unlike in classical planning. However, if $h$ is additionally safe, then admissibility follows from the linear programming formulation of $J^*_*$.

A wide range of SSP heuristic search algorithms exists (e.g., Bonet and Geffner 2003; Trevizan et al. 2017). For all intents and purposes of this work, it suffices to know that these algorithms require an admissible heuristic and compute an optimal policy for a given initial state, if one exists.

Variable Spaces A variable space is a finite set of state variables $V$ associated with non-empty sets $(D(v))_{v \in V}$, called the variable domains. An assignment is an element of $A(V) := \times_{v \in V} D(v)$. A partial assignment is an element of $A^p(V) := \bigcup_{W \subseteq V} \times_{v \in W} D(v)$. For $s, t \in A^p(V)$, we say that $s$ is consistent with $t$, written $s \equiv t$, iff $\text{dom}(s) \supseteq \text{dom}(t)$ and $s(v) = t(v)$ for all $v \in \text{dom}(t)$. We denote with $s[\ell][t] := s[\text{dom}(s) \setminus \text{dom}(t)] \cup t$ the update of $s$ with $t$.

Probabilistic Planning Tasks As the top-level model of the planning problem, we consider probabilistic planning tasks (PPTs) with stochastic effects in finite-domain representation (Trevizan, Thiébaux, and Haslum 2017). A PPT is a tuple $\Pi = \langle V, O, c, s_0, s_f \rangle$. $V$ is a variable space. $O$ is the finite set of operators. Each operator $o \in O$ is associated with a precondition $\text{pre}(o) \in A^p(V)$, and an effect probability distribution $\text{Pr}_o \in D(A^p(V))$. For $s, t \in A^p(V)$, the operator cost function $C(o) := \text{Pr}_o(e)$ if $s \equiv \text{pre}(o)$, else $C(o, s, t) := 0$. The objective is to find an optimal policy for $s_0$ in $\Pi$.

Merge-and-Shrink

We describe merge-and-shrink in terms of the compositional theory by Sievers and Helmut (2021), from which we adopt notation. A (labelled) transition system (TS) is a tuple $\Theta = \langle S, L, c, T_G \rangle$ where $T \subseteq S \times L \times S$ is a transition relation and all other components are the same as for SSPs. A factored transition system (FTS) $F = \langle \Theta_i \rangle_{i \in I}$ is a family of transition systems (called factors) $\Theta_i = \langle S_i, L, c, T_i, G_i \rangle$ with common labels and label cost function. $F$ implicitly represents the synchronized product $O : F := \langle \times_{i \in I} S_i, L, c, T_i, G_i \rangle$ over all of its factors, where $T_i := \langle \{s, t, \ell | \forall i \in I. (s_i, \ell_i, t_i) \in T_i\}$.

The merge-and-shrink algorithm is based on transformations of the represented TS. A transformation of $\Theta$ into $\Theta'$ is a tuple $(\Theta, \Theta', \sigma, \lambda)$ where $\Theta = \langle S, L, c, T_G \rangle$ is the original TS, $\Theta' = \langle S', L', c', T', G' \rangle$ is the transformed TS and $\sigma : S \rightarrow S'$ and $\lambda : L \rightarrow L'$ are state and label mappings that relate the states and labels of both TS.

Transformations on the implicitly represented TS of the FTS are themselves carried out implicitly via factored transformations, which are based on factored mappings (FMs)
which serve as state mappings between FTS. An FM from a variable space $V$ to a set of values $D$ is an inductively defined tree-like data structure. For $v \in V$ and a partial function $\alpha : D(v) \rightarrow D$, $\text{Atom}(v, \alpha)$ is an atomic FM. For FMs $\sigma_t$ from $V_L$ to $D_L$ and $\sigma_R$ from $V_R$ to $D_R$ and a mapping $\alpha : D_L \times D_R \rightarrow D$, $\text{Merge}(\sigma_t, \sigma_R, \alpha)$ is a merge FM. Each FM $\sigma$ from $V$ to $D$ represents a partial function $[\sigma] : A(V) \rightarrow D$. For atomic FMs, $[\text{Atom}(v, \alpha)](s) := \alpha(s(u))$ and for merge FMs, $[\text{Merge}(\sigma_t, \sigma_R, \alpha)](s) := \alpha([\sigma_t](s), [\sigma_R](s))$. For a variable $v \in V$, a projection FM is defined as $\pi_v := \text{Atom}(v, \text{id}_{D(v)})$ with $[\pi_v](s) = s(v)$.

A factored-to-factored mapping (F2FM) from a variable space $V$ to another variable space $V'$ is a family $\Sigma = \{(\sigma_u)_{w \in W} | \text{each } \sigma_u, w \in W \text{ is an FM from } V \text{ to } D(w) \}$ which represents the function $\Sigma : A(V) \rightarrow A(V')$, $s \mapsto (\{\pi_v\}(s))_{w \in W}$. For every two F2FMs $\Sigma$ from $V$ to $W$ and $\Gamma$ from $W$ to $V'$, there is a composed F2FMs $\Gamma \circ \Sigma$ from $V$ to $V'$ with $[\Gamma \circ \Sigma] = [\Gamma] \circ [\Sigma]$.

A factored transformation is a tuple $\langle F, F', \Sigma, \lambda \rangle$ where $F$ is the original FTS, $F'$ is the transformed FTS, $\Sigma$ is an F2FM from $F$ to $F'$, where the variable domain of a factor is its state set, and $\lambda$ is the label mapping. It naturally induces the transformation $\Sigma(F, F', [\Sigma], \lambda)$.

For the purpose of computing admissible heuristics for a given TS $\Theta$, the merge-and-shrink algorithm initially computes a factored representation $F'$ of $\Theta$, the identity F2FM $\Sigma$ from $\Theta$ to $F'$ as well as the identity label mapping $\lambda$ on the labels of $\Theta$. It then repeatedly applies factored transformations on $F$ until reaching a specified limit or until $F$ consists of a single factor. Every factor of $F$ together with $\Sigma$ induces an abstraction heuristic and thus the algorithm can be terminated at any point to return these abstraction heuristics.

There are four standard merge-and-shrink transformations. **Shrinking** applies a state abstraction on some factor $\Theta \in F$. **Merging** replaces two factors $\Theta, \Theta' \in F$ with the factor $\Theta \odot \Theta'$. **Label reduction** applies an abstraction on the common label set of $F$. **Pruning** removes some states and their transitions in some factor $\Theta \in F$.

**Transformations**

Our first objective is the formalization of transformations in the probabilistic setting akin to transformations over transition systems. All proof details for proofs sketched or omitted in the rest of this paper can be found in our technical report (Klößner et al. 2023).

While a TS has a transition relation $T \subseteq S \times L \times S$ and an agent chooses a transition $(s, \ell, t) \in T$ when in state $s$, the SSP model assumes a transition function $P : S \times L \times S \rightarrow [0, 1]$ with $\sum_{s' \in S} P(s, \ell, s') \in \{0, 1\}$ for all $(s, \ell) \in S \times L$, and the agent chooses a label $\ell \in L(s)$. To obtain a proper extension of the classical transformation theory, we first extend the SSP model to adhere to standard TS semantics by extending $P$ to a relation and choosing transitions instead.

**Definition 1 (Probabilistic Transition Systems) A probabilistic transition system (PTS) is a tuple $\Theta = \langle S, L, c, T, G \rangle$ where $T \subseteq S \times L \times D(S)$ is a finite probabilistic transition relation, and all other components are the same as for SSPs. We define $T(s) := \{ (s, \ell, \delta) \in T \}$.**

A policy for $\Theta$ is now a function $\pi : S \rightarrow T$ which terminates or executes a transition $\pi(s) = (s, \ell, \delta) \in T(s)$ when visiting $s$, leading to $t \in S$ with probability $\delta(t)$. The set of $s$-proper policies $\Pi(s)$ and the cost-to-goal functions $J'_0(s)$ and $J'_1(s)$ are defined analogously. The Bellman optimality operator $B_\pi$ now becomes $(B_\pi)(s) := J$ if $T(s) = \emptyset$, else $(B_\pi)(s) := \min_{(s, \ell, \delta) \in T(s)} \{ c(\ell) + \sum_{t \in S} \delta(t) J(t) \}$. As before, a heuristic for $\Theta$ is a function $h : S \rightarrow \mathbb{R}_{\geq 0}$. All heuristic properties are defined analogously.

The induced PTS of a PPT $\Pi = \langle V, \mathcal{O}, c, s_0, s_\pi \rangle$ is the PTS $\Theta(\Pi) = \langle A(V), \mathcal{O}, c, T, G \rangle$ where $G = \{ s \mid s = s_0 \}$ and $T := \{ (s, o, \delta_{s,o}) \mid s \models \text{pre}(o) \}$ with $\delta_{s,o}(t) := \sum_{e \in A^p(v), s[e]=1} \text{Pr}(o)$. Clearly, $J'(S(\Pi)) = J'_0(\Pi(\Theta))$.

We now formally define transformations for PTS. A transformation of $\Theta$ into $\Theta'$ is a tuple $\langle \Theta, \Theta', \sigma, \lambda \rangle$ where $\Theta = \langle S, L, c, T, G \rangle$ is the original PTS, $\Theta' = \langle S', L', c', T', G' \rangle$ is the transformed PTS, $\sigma : S \rightarrow S'$ is a state mapping and $\lambda : L \rightarrow L'$ is a label mapping. The composition of two transformations $r = \langle \Theta, \Theta', \sigma, \lambda \rangle$ and $r' = \langle \Theta', \Theta'', \sigma', \lambda' \rangle$ is the transformation $r \odot r' = \langle \Theta, \Theta'', \sigma \circ \sigma', \lambda \circ \lambda' \rangle$.

To introduce syntactic categories of such transformations, we introduce a natural way to lift a state mapping $\sigma : S \rightarrow S'$ to a state distribution mapping $\sigma_D : D(S) \rightarrow D(S')$ by defining $\text{dom}(\sigma_D) := \{ \delta \mid \text{supp}(\delta) \subseteq \text{dom}(\sigma) \}$ and $\sigma_D(\delta)(s') := \sum_{s \in \text{dom}(\sigma)} \delta(s) \cdot \sigma(s')$. The probability of $s'$ w.r.t. $\sigma_D(\delta)$ is the total probability of states in the preimage of $s'$ w.r.t. $\delta$. This lets us associate any transformation $\tau = \langle \Theta, \Theta', \sigma, \lambda \rangle$ with the mapping $\text{ind}_\tau$ from transitions of $\Theta$ to transitions of $\Theta'$ induced by $\tau$, which is defined as $\text{ind}_\tau((s, \ell, \delta)) := (\sigma(s), \ell, \sigma_D(\delta))$.

With this, we define the following syntactic categories of transformations, which are generalizations of the properties for transformations on transition systems as defined by Sievers and Helmert (2021). We do not consider an explicit initial state, so the properties $\text{IND}_1$ and $\text{CONS}_1$ are omitted.

**Definition 2** Let $\tau = \langle \Theta, \Theta', \sigma, \lambda \rangle$ be a transformation of $\Theta = \langle S, L, c, T, G \rangle$ into $\Theta' = \langle S', L', c', T', G' \rangle$. We define the following list of syntactic properties for $\tau$.

- **CONS$_S$** $\sigma$ is total on $S$
- **CONS$_L$** $\lambda$ is total on $L$
- **CONS$_C$** $\forall \ell \in \text{dom}(\lambda), c'(\lambda(\ell)) \leq c(\ell)$
- **CONS$_T$** $\text{ind}_\tau(T) \subseteq T'$
- **CONS$_G$** $\sigma(G) \subseteq G'$
- **IND$_S$** $\sigma$ is surjective on $S'$
- **IND$_L$** $\lambda$ is surjective on $L'$
- **IND$_C$** $\forall \ell' \in L', \exists \ell \in \lambda^{-1}(\ell'). c(\ell) = c'(\ell')$
- **IND$_T$** $\text{ind}_\tau(T) \supseteq T'$
- **IND$_G$** $\sigma(G) \supseteq G'$
- **REF$_C$** $\forall \ell' \in L', \forall \ell \in \lambda^{-1}(\ell'). c(\ell) = c'(\ell')$
- **REF$_T$** $\forall s' \in S', T'(s') \subseteq \bigcup_{s \in \sigma^{-1}(s')} \text{ind}_\tau(T(s))$
- **REF$_G$** $\sigma^{-1}(G') \subseteq G$

We say that $\tau$ is conservative (an abstraction) iff $\tau$ satisfies $\text{CONS}_{\text{TL+C+T}} \land \text{IND}_{\text{TL+C+T}} \land \text{REF}_{\text{C+T}}$. Furthermore, we say that $\tau$ is exact iff it is conservative and refinable.
All properties listed in Definition 2 are semantically equivalent to their respective counterpart in the classical theory, except for IND\textsubscript{T}, CONS\textsubscript{T} and REF\textsubscript{T}, which are generalizations for stochastic transitions. If every transition is deterministic, i. e., has a unique successor with probability one, then these properties can also be considered equivalent.

As in the classical case, for every PTS Θ and total and surjective state and label mappings σ and λ, there is a unique PTS Θ\textsuperscript{σ,λ} induced by σ and λ such that \( (Θ, Θ\textsuperscript{σ,λ}, σ, λ) \) is induced conservative. An example is depicted in Figure 1.

Like their classical counterparts, all of these syntactic properties are compositional. If any of the above properties is shared individually by two composed transformations, the property is inherited by the composition.

Theorem 1 Let τ and τ′ satisfy one of the properties of Definition 2. Then τ′ ◦ τ also satisfies this property.

Proof. All properties except CONS\textsubscript{T}, IND\textsubscript{T} and REF\textsubscript{T} are semantically equivalent in the classical theory, where they compose. Let τ = (Θ, Θ′, σ, λ) and τ′ = (Θ′, Θ″, σ′, λ′). Let T, T′ and T′′ be the transitions of Θ, Θ′ and Θ″.

Regarding CONS\textsubscript{T}, we have ind\textsubscript{T}(T) ⊆ T′ due to CONS\textsubscript{T} for τ, hence ind\textsubscript{T′′−r}(T′′) = ind\textsubscript{T′′−r}(ind\textsubscript{T′}(T′)) ≤ ind\textsubscript{T′′−r}(T′′) ⊆ T′′ where the first subsumption involves the image of a superset under ind\textsubscript{T′}, and the last subsumption follows from CONS\textsubscript{T} for τ′. For IND\textsubscript{T}, the proof is identical, except that we replace ⊆ and CONS\textsubscript{T} with ⊇ and IND\textsubscript{T}.

For REF\textsubscript{T}, let s″ ∈ Θ″. By REF\textsubscript{T} for τ′ and τ, we have

\[
T′′(s″) ⊆ \bigcup_{s′ ∈ σ′−1(s″)} \bigcup_{s′′ ∈ σ″−1(s′′)} \text{id}_{s′′}(\bigcap_{s″ ∈ σ″−1(s′′)} T(s″))
\]

where subsumption two involves the image of a superset and subsumption three is basic set algebra.

For any transformation τ = (Θ, Θ′, σ, λ), we define a corresponding heuristic \( h_{Θ}^{τ}(s) := J_{Θ}^{σ}(σ(s)) \) if \( s ∈ \text{dom}(σ) \) and \( h_{Θ}^{τ}(s) := \infty \) otherwise for the TS Θ, as in the classical theory. Additionally, we define a corresponding heuristic for Θ′ by \( h_{Θ}^{τ}(s′) := \max_{s ∈ σ′−1(s′)} J_{Θ}^{σ}(σ(s)) \) if \( σ′−1(s′) \neq \emptyset \), else \( h_{Θ}^{τ}(s′) := 0 \). Depending on the syntactic category of the transformation, we can derive guarantees for these heuristics which are similar to those observed in the classical setting.

Theorem 2 Let τ = (Θ, Θ′, σ, λ) be a transformation. (i) If τ is conservative, then \( h_{Θ}^{τ} \) is goal-aware, consistent, safe and admissible. (ii) If τ is refinable, then \( h_{Θ}^{τ} \) is goal-aware, consistent, safe and admissible. (iii) If τ is refinable, then \( h_{Θ}^{τ} \) is pessimistic. (iv) If τ is exact, then \( h_{Θ}^{τ} \) is perfect.

Proof (sketch). (iii) follows from (ii) since \( h_{Θ}^{τ}(s) = J_{Θ′}(σ(s)) \geq J_{Θ′}(σ(s)) \geq J_{Θ}(σ(s)) \) by admissibility if \( s ∈ \text{dom}(σ) \) and \( h_{Θ}^{τ}(s) = \infty \geq J_{Θ}(σ(s)) \) otherwise. Clearly, (iv) follows directly from (i) and (iii). Admissibility of the heuristics follows from goal-awareness, consistency and safety.

Goal-awareness of \( h_{Θ}^{τ} \) (respectively \( h_{Θ}^{τ} \)) follows directly from CONS\textsubscript{S} and CONS\textsubscript{C} (respectively REF\textsubscript{C}).

Consistency of \( h_{Θ}^{τ} \) is shown by proving the inequality \( h_{Θ}^{τ}(s) = J_{Θ}(σ(s)) \geq h_{Θ′}(σ(s)) \geq J_{Θ′}(σ(s)) \) by admissibility if \( s ∈ \text{dom}(σ) \) and \( h_{Θ}^{τ}(s) = \infty \geq J_{Θ}(σ(s)) \) otherwise. Consistency of \( h_{Θ}^{τ} \) is shown by proving \( J_{Θ}(σ(s)) \leq \max_{s ∈ \text{dom}(σ)} J_{Θ}(σ(s)) \) for all states \( s ∈ \Theta \).

Regarding safety of \( h_{Θ}^{τ} \), we argue that the maximal goal probability of a state \( s ∈ Θ \) is lower or equal to the maximal goal probability of the state \( σ(s) ∈ Θ′ \). Therefore, \( J_{Θ}(σ(s)) \leq J_{Θ′}(σ(s)) \) holds. For safety of \( h_{Θ′} \), we show that the maximal goal probability of a state \( s′ ∈ Θ′ \) is lower bound for the maximal goal probability of all states \( s ∈ σ−1(s′) \) in Θ′. This means that \( J_{Θ′}(s′) \leq J_{Θ}(σ(s)) \) holds. Therefore, we have \( J_{Θ′}(s′) = \max_{s ∈ σ−1(s′)} J_{Θ}(σ(s)) \) for all \( s ∈ \text{dom}(σ) \).

Beyond these basic properties, we can also pinpoint exactly when an induced and conservative transformation is also refinable. In classical planning, bismulations can be used to characterize this property for transformations which only alter the states. We now revisit this property in our setting, this time formulated in terms of a variant of probabilistic bismulation (Larsen and Skou 1991). For probabilistic transition systems, we define bismulation as follows:

Definition 3 (PTS Bismulation) Let \( Θ = (S, L, c, T, G) \) be a PTS. A PTS bismulation on Θ is an equivalence relation \( ∼ \subseteq S \times S \) such that for all states \( s_1, s_2, s \in S \) with \( s_1 ∼ s_2 \): (BISM\textsubscript{M}_{1}) if \( s_1 ∈ G \) then \( s_2 ∼ s \) and (BISM\textsubscript{M}_{1}) if \( s_1, t, δ_1, δ_2, δ \in T \) with \( δ_1 ∼ δ_2, δ \), where \( δ_1 ∼ δ_2 \) if and only if \( \sum_{s ∈ C} δ(s) = \sum_{s ∈ C} δ′(s) \) for all equivalence classes \( C ∈ S/∼ \).
Definition 4 (Atomic PTS Projections) Let $\Pi = (V, O, c, s_0, s_*)$ be a PPT. The atomic projection to $v \in V$ is defined as $\Theta^v(\Pi) := (D(v), O, c, T^v, G^v)$ where $G^v := D(v)$ if $v \notin \text{dom}(s_*)$ and $G^v := \{s_*(v)\}$ otherwise, and $T^v := \{(d, o, \delta_{d,o}) \mid v \notin \text{dom}(\text{pre}(o)) \text{ or } d = \text{pre}(o)\}$ where $\delta_{d,o}(d') := \sum_{e \in D(v), (v \notin \text{dom}(e)) \wedge d' = d'' \wedge e(v) = d'} \text{Pr}_o(e)$.

This definition generalizes the definition of the transition relation in the most natural way while keeping all other components the same as in the traditional framework. In classical planning, $\Theta(\Pi)$ can be reconstructed from the atomic projections by computing the synchronized product. Therefore, the atomic projections yield a suitable initialization of the factored TS maintained during the algorithm. Regrettably, this guarantee is lost in presence of stochastic operators.

As an example, consider a parameterized PPT $\Pi_p$ with $p \in [0, \frac{1}{4}]$. $\Pi_p$ has two boolean variables $v, w \in V$ and a single operator $o \in O$. Denoting the effect $\{v \rightarrow x, w \rightarrow y\}$ with $e_{xy}$, $o$ has the precondition $\text{pre}(o) = e_{00}$ and effect probabilities $\text{Pr}_o(e_{00}) = p$, $\text{Pr}_o(e_{01}) = \frac{1}{4} - p$, and $\text{Pr}_o(e_{11}) = \frac{1}{2} + p$. The induced PTS $\Theta(\Pi_p)$ is depicted in Figure 2a and the atomic projections $\Theta^v(\Pi_p)$ and $\Theta^w(\Pi_p)$ are depicted in Figure 2b, where transitions are also annotated with the effects that contribute a positive probability. These annotations are not part of the state space model and are only depicted for demonstrative purposes. As we can see, $\Theta^v(\Pi_p)$ and $\Theta^w(\Pi_p)$ are completely independent from the parameter $p$, whereas $\Theta(\Pi_p)$ is not. Therefore, atomic projections do not contain enough information to uniquely determine and thus reconstruct the induced PTS of a planning task, which makes the definition of a reconstructing synchronized product impossible.

The problem is that we lose information about the individual outcomes of the operators when we build an atomic projection. If we want to synchronize the atomic projections correctly, we must not only make sure that the same operator is applied in all atomic projections, but also that the outcomes are synchronized and happen with the same probability. Regrettably, a transition in an atomic projection is still associated with its original operator that induced it, but can correspond to multiple distinct outcomes with an identical effect on the projection variable. The individual probabilities of these outcomes are lost, so synchronization is impossible.

In order to synchronize on the individual operator outcomes, we henceforth keep track of them in our underlying state space model. To this end, we integrate a set of label effects $E$, which are opaque identifiers for the outcomes of an operator and annotate the transitions with them accordingly.

The probabilities are now encoded in the label effects instead of the transitions. Thus, we only store a successor mapping.

![Figure 2: The induced PTS and atomic (A)PTS projections of the example planning task $\Pi_p$, where $p \in [0, \frac{1}{4}]$.](image)
\[ \alpha : E \rightarrow S \] from label effects to corresponding successors. Given a transition \( \langle s, \ell, \alpha \rangle \), the intuition is that effect \( e \) occurs with probability \( D_\ell(e) \) and leads to state \( \alpha(e) \). If \( D_\ell(e) = 0 \), the associated successor is impossible and could be formally omitted by using partial mappings instead. We choose not to do so to keep the notation simple.

Of course, in the context of a transition with label \( \ell \), we can always cast a successor mapping \( \alpha : E \rightarrow S \) to a successor distribution \( \delta_\ell(\alpha) \in D(S) \) defined as \( \delta_\ell(\alpha)(e) := \sum_{I \in \alpha^{-1}(\ell)} D_\ell(e) \), dropping the additional information. Doing this for every transition, we obtain the natural interpretation as a PTA. Formally, given \( \Pi = (V, O, c, s_0, s_s) \) in the natural way. The label-effects \( E := \bigcup_{o \in O} \text{supp}(P_{ro}) \) correspond to the possible effects of the operators on the state variables, the label effect probabilities \( D_o(e) := \Pr_{ro}(e) \) match with the cost function coincide because they are unaffected by the product. The argument for the (goal) states matches the classical proof. Before we focus on the transitions, we note that \( \alpha_s = \bigtimes_{v \in V} \alpha_{s_v} \) for all states \( s \in A(V) \).

\[ \begin{align*}
\text{“C” Let } & \langle s,o,\alpha \rangle \in A(\Pi). \text{ By definition of } A(\Pi), \alpha = \alpha_s \text{ and } s = \text{pre}(o). \text{ Hence, we have } s_o = \text{pre}(o), \text{ for all } v \in \text{dom}(\text{pre}(o)). \text{ By definition of the } A^v(\Pi), \langle s_v,o,\alpha_{s_v} \rangle \in A^v(\Pi) \text{ for all } v \in V. \text{ By definition of the product, we have } \langle s,o,\bigtimes_{v \in V} \alpha_{s_v} \rangle = (s,o,\alpha_s) \in \bigtimes_{v \in V} A^v(\Pi). \\
\text{“D” Let } & \langle s,o,\alpha' \rangle \in \bigtimes_{v \in V} A^v(\Pi). \text{ By definition of the product, there is } \alpha \rightarrow \alpha' \in \bigtimes_{v \in V} A^v(\Pi) \text{ such that } \langle s_v,o,\alpha_{s_v} \rangle \text{ and } s_v = \text{pre}(o) \text{ for all } v \in \text{dom}(\text{pre}(o)). \text{ Ultimately, } s = \text{pre}(o) \text{ and } \langle s,o,\alpha_s \rangle = (s,o,\bigtimes_{v \in V} \alpha_{s_v}) \in A(\Pi). \quad \square
\end{align*} \]

**Merge-and-Shrink Transformations**

As we have just seen, APTSs are expressive enough to formulate an appropriate synchronous product. We will now use this state space model as a foundation to formally describe our merge-and-shrink framework for probabilistic planning.

As the underlying factored representation of the state space, we maintain a family of APTS \( F = (\Pi_i)_{i \in I} \) where \( I \) is a finite index set. \( F \) implicitly represents the APTS \( \bigtimes F \). We call \( F \) a factored APTS (FAPTS) and its elements the factors of \( F \). All factors share the same labels, label effects, label effect probabilities and cost function. Given a task \( \Pi \) with variables \( V \), we start with the initialization \( F = (A^v(\Pi))_{v \in V} \), as justified by Theorem 4. Clearly, each \( A^v(\Pi) \) is constructible in time polynomial in the size of \( \Pi \).

To define the basic merge-and-shrink transformations, we first introduce APTS transformations in the same way as for probabilistic transition systems. A transformation of APTS \( A \) with states \( S \) and labels \( L \) into \( A' \) with states \( S' \) and labels \( L' \) is a tuple \( \tau = (A,A',\sigma,\lambda) \) where \( A \) is the original APTS, \( A' \) is the transformed APTS and \( \sigma \) and \( \lambda \) are again state and label mappings. The composition \( \tau' \circ \tau \) of two APTS transformations \( \tau \) and \( \tau' \) is defined analogously. The FAPTS \( F \) is subject to factored transformations into some other FAPTS \( F' \), analogously defined as tuples \( (F,F',\Sigma,\lambda) \) where \( \Sigma \) is an F2FM. Such a factored transformation induces the non-factored transformation \( \bigtimes (F,F',[\Sigma],\lambda) \).

The main purpose of the APTS model is to enable synchronization of the factors and hence to enable a definition of merge transformations with appropriate guarantees, as we will see. When we want to compute a heuristic from a factor \( A \), we do however not need the additional information and compute the heuristic values from the induced PTS \( \Theta(A) \). Therefore, for an APTS transformation \( \tau = (A,A',\sigma,\lambda) \), we define the induced heuristic \( h^\tau_A(s) := h^{PTS}_{\Theta(A)}(s) \), where \( \tau_{PTS} := (\Theta(A),\Theta(A'),\sigma,\lambda) \) is the induced PTS transformation of \( \tau \). Since we define \( h^\tau_A \) directly in terms of \( \tau_{PTS} \), we simply say that \( \tau \) satisfies a syntactic property of Definition 2 iff \( \tau_{PTS} \) satisfies it and obtain the equivalent of Theorem 2 for APTS transformations.

**Theorem 4** Let \( \Pi \) be a probabilistic planning task with variables \( V \). Then \( A(\Pi) = \bigtimes_{v \in V} A^v(\Pi) \).

**Proof.** The labels, label effects, label effect probabilities and the cost function coincide because they are unaffected by the product. The argument for the (goal) states matches the classical proof. Before we focus on the transitions, we note that \( \alpha_s = \bigtimes_{v \in V} \alpha_{s_v} \) for all states \( s \in A(V) \).
We now carry over three of the four factored transformations used in merge-and-shrink — Shrinking, Merging, Label Reduction — to our setting and show that fundamental properties still apply. We also investigate more advanced instantiations of these transformations which yield stronger guarantees for the constructed merge-and-shrink heuristic.

Shrink Transformations

Shrink transformation are conceptually unchanged. A shrink transformation applies some local abstraction on one of the factors. To this end, let $A = \langle S, L, c, E, D, T, G \rangle$ be an APTS and let $\sigma: S \rightarrow S'$ be a total state mapping. We will denote with $A^\sigma$ the APTS $\langle S', L, c, E, D, T', \sigma(G) \rangle$ where $T' := \{(s, \ell, \sigma \circ \alpha) \mid (s, \ell, \alpha) \in T\}$ in the following.

Definition 8 (Shrink Transformation) Let $F = (A_i)_{i \in I}$ be a FAPTS and let $\sigma$ be a total state mapping on some $A_k \in F$. The shrink transformation for $A_k$ is the factored transformation $(F, F', \Sigma, \text{id})$ where $F' = (A_i)_{i \in I \setminus \{k\}} \cup \{k \mapsto A_k^\sigma\}$ and $\Sigma = (\pi_i)_{i \in I \setminus \{k\}} \cup \{k \mapsto \text{Atom}(k, \sigma)\}$.

In the classical merge-and-shrink framework, shrink transformations are generally induced and conservative. Fortunately, this property naturally extends to our framework.

Theorem 5 Shrink transformations are induced abstractions.

Proof (sketch). The proofs for the properties $\text{CONS}_{S+L+C+G}$ and $\text{IND}_{S+L+C+G}$ are identical in the classical theory. For $\text{CONS}_T$ and $\text{IND}_T$, we extend the corresponding classical proof for transition systems in a straightforward manner by substituting every occurrence of a successor state with a successor mapping. □

An interesting classical class of shrinking strategies is based on bisimulation (Milner 1990). Bisimulation-based shrinking strategies are prominent in classical planning (Nissim, Hoffmann, and Helmer 2011) because they yield exact transformations. In Theorem 3, we have shown a similar relationship between PTS bisimulations and exact transformations. The natural extension of this strategy is then to shrink states in the selected factor $A_k \in F$ according to a PTS bisimulation on $\Theta(A_k)$. Unfortunately, unlike in classical planning, this transformation is not exact, as such a shrink transformation may not correspond to a PTS bisimulation on $\otimes F$. A counter-example is depicted in Figure 3.

The problem is that in the APTS $A_1$ in Figure 3, the transitions with label $\ell_1$ have bisimilar successor distributions in $\Theta(A_1)$, but do not lead to bisimilar states for a fixed label effect. By adding this requirement, we could recover exactness. However, we will consider a slightly more general approach. We say that two label effects $e_1$ and $e_2$ are locally equivalent for label $\ell$ in $A$ if $(s, \ell, \alpha_1) \in A$ implies $\alpha(e_1) = \alpha(e_2)$. In the context of a FAPTS $F$, we say that two label effects $e_1$ and $e_2$ are $A$-combinable for $\ell$ if they are locally equivalent for $\ell$ in every other factor $A' \in F \setminus \{A\}$. We write $e_1 \simeq_{A, \ell} e_2$ when $e_1$ and $e_2$ are $A$-combinable for $\ell$. Note that $\simeq_{A, \ell}$ is an equivalence relation. Now, instead of enforcing bisimilar successors for each label effect, we only require that the successor distributions are bisimilar under the condition that a fixed but arbitrary class of $A$-combinable effects is triggered. This leads to the following definition.

Definition 9 (APTS Bisimulation) Let $F$ be a FAPTS. An APTS bisimulation on a factor $A = \langle S, L, c, E, D, T, G \rangle$ of $F$ is an equivalence relation $\sim \subseteq S \times S$ such that for all $s_1, s_2 \in S$ with $s_1 \sim s_2$, (i) if $s_1 \in G$ then $s_2 \in G$ and (ii) if $(s_1, \ell, \alpha_1) \in T$ then there is $(s_2, \ell, \alpha_2) \in T$ with $\alpha_1 \sim \alpha_2$, where $\alpha_1 \sim \alpha_2$ iff $\sum_{e \in D_r\alpha^{-1}_1(c)} D_t(e) = \sum_{e \in D_r\alpha^{-1}_2(c)} D_t(e)$ for all equivalence classes $C \in S/\sim$ and $D \in E/\simeq_{A, \ell}$.

Theorem 6 APTS bisimulation-based shrinking is exact.

Proof (sketch). Let $\tau = (F, F', \Sigma, \text{id})$ be a shrink transformation for $\sigma$ and $A_k \in F$. The idea is to show that $\sim_{\tau[\Sigma]}$ is a PTS bisimulation on $\Theta(\otimes F)$ if $\sim_{\sigma}$ is an APTS bisimulation on $A_k$. Exactness then follows by Theorem 3. □

Merge Transformations

Merge transformations are also conceptually identical to their classical variant. They replace two factors by their synchronous product.

Definition 10 (Merge Transformation) Let $F = (A_i)_{i \in I}$ be a FAPTS and let $j, k \in I, j \neq k$. The merge transformation for $j$ and $k$ is the factored transformation $(F, F', \Sigma, \text{id})$ where $F' := (A_i)_{i \in I \setminus \{j, k\}} \cup \{(j, k) \mapsto A_j \otimes A_k\}$ and $\Sigma := (\pi_i)_{i \in I \setminus \{j, k\}} \cup \{(j, k) \mapsto \text{Merge}(\pi_j, \pi_k, \text{id})\}$. 

Figure 3: A counter-example illustrating inexactness of PTS bisimulation-based shrinking. Here, both labels have unit cost. The single effect of $\ell_2$ is not explicitly drawn. On the left, two factors $A_1$ and $A_2$ and on the right, their implicitly represented APTS $A_1 \otimes A_2$ are depicted. While $s_1$ and $s'_1$ are clearly bisimilar in $\Theta(A_1)$, shrinking them would cause the states $s_1 s_2$ and $s'_1 s_2$ to be merged in $A_1 \otimes A_2$. This would decrease the optimal cost-to-goal of $s'_1 s_2$ in $A_1 \otimes A_2$ from 2 to 1.5.
In classical planning, a merge transformation is not only exact, but $\otimes F$ and $\otimes F'$ are even identical up to renaming of states. Transformations with this property are also called *isomorphisms*. To show the analogous property for our setting, we define isomorphisms on APTS as transformations $\tau = (A, A', \sigma, \text{id})$ such that $\sigma$ is bijective. It is straightforward to see that isomorphisms are composable, exact transformations as in classical planning.

**Theorem 7** Merge transformations are isomorphisms.

**Label Reduction**

Lastly, we consider label reduction. In contrast to classical label reduction, it is harder to give a general definition for this transformation in our setting, since we also need to explain how to unify the label effects of the labels, and also unify their probabilities. Remember that a label effect is just a token that identifies a possible outcome for a specific label. The alphabet of tokens is shared across labels, but the semantics of a label effect in only defined in the context of a label, and is independent of other labels.

For label reductions, we assume a labeling mapping $\lambda: L \rightarrow L'$ that maps each label $\ell \in L$ of the FAPTS $F$ to a reduced label $\lambda(\ell) \in L'$. To be able to interpret an outcome of $\ell$ as an effect of the reduced label $\lambda(\ell)$, we additionally parameterize the reduction by a family of label effect mappings $\epsilon = (\epsilon_i)_{i \in I}$. Each $\epsilon_i: E \rightarrow E'$ bijectively maps from the old label effects $E$ of $F$ to a set of new label effects $E'$. We say that $\epsilon$ unifies label $\ell_1$ and $\ell_2$ iff for every $\epsilon_1, \epsilon_2 \in E$, $\epsilon_1(\ell_1) = \epsilon_2(\ell_2)$ implies $D_{\ell_1}(\epsilon_1) = D_{\ell_2}(\epsilon_2)$, i.e., label effects that map to the same new label effect have the same probability. Given a labeling mapping $\lambda$, we say that $\epsilon$ is a unifier for $\lambda$ iff $\lambda(\ell_1) = \lambda(\ell_2)$ implies $\epsilon$ unifies $\ell_1$ and $\ell_2$.

Now, given the total and surjective labeling mapping $\lambda: L \rightarrow L'$ and a unifier $\epsilon = (\epsilon_i)_{i \in I}$ for $\lambda$ with $\epsilon_i: E \rightarrow E'$, we define the label reduced APTS of $A$ by $A^{\lambda, \epsilon} := \langle \{L, E', E', D', T', G\} | D'_{\ell} := D_{\ell} \circ \epsilon_1^{-1} \text{ for any } \ell \in \lambda^{-1}(E') \rangle$. Moreover, $c'(\ell') := \min_{\ell \in \lambda^{-1}(\ell')} c(\ell)$ and $T' := \{ (s, \lambda(\ell), \alpha \circ \epsilon_1^{-1}) \mid (s, \ell, \alpha) \in T \}$. With this, label reduction transformations are defined as follows.

**Definition 11 (Label Reduction)** Let $F = (A_i)_{i \in I}$ be a FAPTS with labels $L$. Let $\lambda$ be a total, surjective labeling mapping defined on $L$ and let $\epsilon$ be a labeling effect unifier for $\lambda$. The label reduction transformation for $\lambda$ and $\epsilon$ is the factored transformation $\langle F, (A_i^{\lambda, \epsilon})_{i \in I}, (\pi_i)_{i \in I}, \lambda \rangle$.

We can show that our formalization of label reduction satisfies the same syntactic properties as in the classical case.

**Theorem 8** Label reductions are abstractions and satisfy IND$_{\lambda \epsilon C \Gamma}$ and REF$_{\Gamma}$. Furthermore, they satisfy REF$_{\mathcal{C}}$ if and only if only labels with the same costs are reduced and satisfy IND$_{T}$ if and only if they satisfy REF$_{\mathcal{T}}$.

**Proof (sketch).** The proofs for properties CONS$_{\lambda \epsilon C \Gamma}$, IND$_{\lambda \epsilon C \Gamma}$ and REF$_{\Gamma}$, and the necessary and sufficient condition of REF$_{\mathcal{C}}$ are given in the classical theory. Since $[\Sigma] = \text{id}$ we have $\bigcap_{s \in \Sigma} \text{ind}_{T}(T(s)) = \text{ind}_{T}(s')$ and so IND$_{T}$ and REF$_{T}$ collapse to a common statement.

**For CONS$_{T}$**, let $\langle s, \ell, \delta \rangle \in \Theta(\otimes F)$. By definition of $\Theta(A)$ and the product, there is some $\alpha$ such that $\delta = \delta \otimes (\otimes_{i \in I} \alpha_i)$ and $\langle s_i, \ell_i, \alpha_i \rangle \in A_i$ for all $i \in I$. By definition of $A^{\lambda, \epsilon}$, we get $\langle s_i, \lambda(\ell), \alpha_i \circ \epsilon_1^{-1} \rangle \in A^{\lambda, \epsilon}$ for all $i \in I$. By definition of the product and $\Theta(A)$, $\langle s, \lambda(\ell), \delta \lambda(\ell) (\otimes_{i \in I} \alpha_i \circ \epsilon_1^{-1}) \rangle \in \Theta(\otimes F')$. It is left to show that $\delta \lambda(\ell) (\otimes_{i \in I} \alpha_i \circ \epsilon_1^{-1}) \subseteq \Theta(\otimes F')$. Due to space reasons, we give a proof of this claim in the technical report.

Last but not least, we propose an exact label reduction strategy. To this end, we only consider atomic label reductions, which only reduce exactly two labels. Recall that in classical planning, it is NP-hard to decide whether a non-atomic label reduction is exact, however, exactness of atomic label reductions can be characterized in terms of the notions of label subsumption and $\Theta$-combinability. We will extend these notions to our setting by integrating the label effect mapping $\epsilon$ of a label reduction into these concepts.

For an APTS $A$, a label $\ell$ of $A$ and a unifier $\epsilon$ for $\lambda$, we define $T_{\epsilon}(A, \ell) := \{ (s, \alpha \circ \epsilon_1^{-1}) \mid (s, \ell, \alpha) \in A \}$. If there is no transition $\langle s, \ell, \alpha \rangle \in A$, then we say $\ell$ is dead in $A$. We say that $\ell_1 \epsilon$-subsumes $\ell_2$ in $A$ iff $\epsilon$ unifies $\ell_1$ and $\ell_2$ and if $T_{\epsilon}(A, \ell_1) \subseteq T_{\epsilon}(A, \ell_2)$. If $\epsilon$-subsumption holds in both directions, we say $\ell_1$ and $\ell_2$ are $\epsilon$-equivalent in $A$. Lastly, given a FAPTS $F$ we say $\ell_1$ and $\ell_2$ are $(A, \epsilon)$-combinable iff $\ell_1$ and $\ell_2$ are $\epsilon$-equivalent in every other factor $A' \in F \setminus \{ A \}$. With this, we state the following sufficient condition for exactness of atomic label reduction. In contrast to its classical counterpart, we do not claim that this condition is necessary.

**Theorem 9** A label reduction of a FAPTS $F$ that combines exactly two labels $\ell_1, \ell_2$ of $F$ is exact if $c(\ell_1) = c(\ell_2)$ and either (A) $\ell_1$ $\epsilon$-subsumes $\ell_2$ in all $A \in F$ or vice versa, (B) $\ell_1$ and $\ell_2$ are $(A, \epsilon)$-combinable for some $A \in F$ or (C) $\ell_1$ and $\ell_2$ are dead for some $A \in F$.

**Proof (sketch).** The proof is an extension of the corresponding exactness proof in the classical theory. First, we show that each of the conditions (A), (B) and (C) imply the subsumption

$$\bigtimes_{i \in I} \bigcup_{\ell \in \lambda^{-1}(\ell')} T_{\epsilon}(A_i, \ell) \subseteq \bigcup_{\ell \in \lambda^{-1}(\ell')} \bigtimes_{i \in I} T_{\epsilon}(A_i, \ell).$$

In a second step, we can then show that this inclusion is sufficient to prove IND$_{T}$ by contraposition. Exactness then follows directly from Theorem 8.

**Conclusion**

We presented an extension of Merge-and-Shrink as a compositional theory of transformations tailored for probabilistic planning. We have shown that our extension properly generalizes important properties of the classical theory and provides a new way of constructing admissible SSP abstraction heuristics. Possible future work in this direction includes a formalization of pruning transformations, as well as an experimental evaluation of Merge-and-Shrink as a construction technique for admissible SSP heuristics.
Acknowledgements

This work was funded by DFG grant 389792660 as part of TRR 248 (see https://perspicuous-computing.science). Silvan Sievers was partially supported by TAILOR, a project funded by the EU Horizon 2020 research and innovation programme under grant agreement no. 952215. He further has received funding for this work from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement no. 817639).

References


Kolobov, A.; Mausam; and Weld, D. S. 2012. Discovering Hidden Structure in Factored MDPs. Artificial Intelligence, 189: 19–47.


