Conditional Dominance Analysis for Classical Planning

Anna Wilhelm[a] **and Álvaro Torralba** [b,*]

[a]Saarland University, Saarland Informatics Campus, Saarbrücken, Germany
[b]Aalborg University, Aalborg, Denmark

Abstract. Dominance analysis methods compare pairs of states in a planning task to prove that one is at least as close to the goal as other. Existing methods compute fact-dominance relations, which identify facts that are at least as good as others in any situation. However, this is only possible when a fact is at least as good as another in every single possible context. We introduce a new notion of conditional dominance, which can identify that a fact dominates another under certain conditions. We extend previous methods to compute dominance by taking into account a set of "contexts" in order to find maximal dominance relations. We propose several strategies to find relevant contexts automatically and show that even with one single condition, one can achieve significant pruning in certain domains.

1 Introduction

Dominance analysis is concerned with identifying when one state is guaranteed to be at least as close to the goal as another [17, 13]. This has many applications in the context of automated planning [5] such as pruning dominated nodes [17, 15], deriving contrastive heuristics [12], eliminating irrelevant actions [18], ranking states [14], and policy testing [4]. In principle, one could determine dominance by directly computing the goal distances of both states and comparing them. However, computing exact goal distances is generally intractable. The central question, then, is under which conditions dominance can be established efficiently (in polynomial time) without necessarily computing goal distances. For instance, if states differ only in a resource such as energy or fuel, the state with more resources dominates, since additional resources can only increase available actions.

Existing dominance methods are compositional: they construct a relation for each component of the planning task (often called a state variable). These relations compare alternative values of a variable (also called facts) and identify when one value is always at least as good as another, in the sense that replacing it can never increase the goal distance. A state dominates another if this holds for all variables. This goes beyond identifying resource variables such that having more (or less) is always better. For example, in a logistics task where the goal is to deliver several packages using a truck, having a package already at the destination is always at least as good as having it at the origin.

Compositional methods are very powerful because, after identifying dominance among facts, this can be used to prove that exponentially many states dominate other states (e.g. all states that only differ on that fact). However, this is also a limitation because, for a fact to dominate another, it needs to be as good in all possible contexts. In some tasks, this simply does not hold for any pair of distinct facts, rendering current dominance analysis methods entirely uninformative. This can be alleviated by considering sets of variables at once [17]. However, the runtime of dominance analysis grows exponentially with the number of variables. And sometimes, there are dependencies on most variables (e.g. in the example above which position is preferable for the truck depends on where all packages are).

We introduce conditional fact-dominance, where one fact dominates another but only under specific restrictions on the values of other variables. We represent restrictions as per-variable relations, specifying which pairs of values may appear in the states being compared. This is very flexible, as it does not require the two states to share the same value. Furthermore, a condition considers the value of all variables, addressing the main limitation of previous methods. We show that, given a predefined set of conditions, conditional dominance relations can be computed in polynomial time. Moreover, the framework leverages synergy across multiple conditions, so that fact-dominance under multiple conditions can strengthen one another.

The result is a highly expressive framework for defining dominance relations, capable of identifying strong dominances in planning tasks and domains where previous methods fail to find any. We demonstrate this both theoretically — by showing conditions under which our approach discovers powerful dominances in a running example where unconditional dominance yields no pruning — and empirically, as our method is the first to detect dominance in certain IPC domains such as Blocksworld. However, selecting useful condition sets is challenging, as their number grows doubly exponentially with the number of variables. As a first step, we introduce methods to automatically generate single conditions. While these strategies do not yet exploit the full potential of the framework, they already yield informative dominance across several domains.

2 Background

A *transition system* (TS) is a tuple $\Theta = \langle S, L, c, T, s^I, S^G \rangle$ where S is a finite set of *states*, L is a finite set of *labels*, $c : L \mapsto \mathbb{Q}_0^+$ is a label cost function, $T \subseteq S \times L \times S$ is a set of *transitions*, $s^I \in S$ is the *initial state*, and $S^G \subseteq S$ is the set of *goal states*. We use $s \xrightarrow{\ell} t$ as a shorthand for $(s, \ell, t) \in T$. A *path* $\pi = s \xrightarrow{\ell_1} \dots \xrightarrow{\ell_k} t$ from s to $t \in S$ is a sequence of $k \geq 1$ transitions $(s_{i-1}, \ell_i, s_i) \in T$ for $i \in \{1, \dots, k\}$, starting in $s = s_0$ and ending in $t = s_k$. The cost of π is $c(\pi) := \sum_{i \in \{1,\dots,k\}} c(\ell_i)$. The empty path is a path from s to s, and has length $k = 0$ and cost 0. A *plan* for s is a path from s to any $s' \in S^G$. The minimum cost of any plan for s is denoted by $h^*(s)$. A plan for s is *optimal* if its cost equals $h^*(s)$.

[*] Corresponding Author. Email: alto@cs.aau.dk

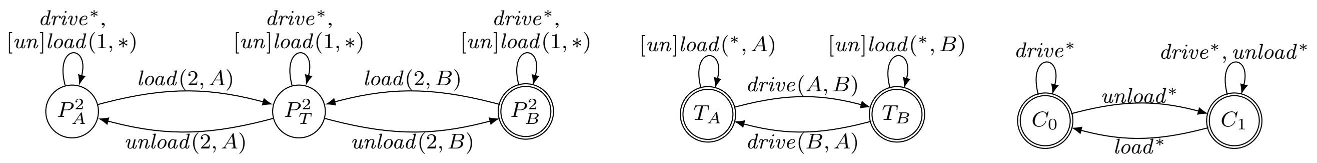

Figure 1. Factors representing the position of a package v_{p_2}, the truck v_t, and the capacity v_c of our running example.

2.1 *Representation of Planning Tasks*

We consider planning tasks in a factored representation, where states are described in terms of finite-domain variables, and each variable (also called factor) is described as a TS [9, 19, 11, 2].

Definition 1. *A factored task $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n \rangle$ is a tuple of transition systems with a common set of labels, L, and cost function c, such that $\Theta_i = \langle S_i, L, c, T_i, s_i^I, S_i^G \rangle$ for $i \in \{1, \dots, n\}$.*

We use a tuple instead of a set to refer to the individual factors by their index. To differentiate states in a factored task and in each of the factors, Θ_i, we refer to the states in the individual factors $s_i \in S_i$ as *facts*. A state in the planning task is a tuple of facts $s = \langle s_1, \dots, s_n \rangle$, one for each factor, $s_i \in S_i$. The state space of a factored task is the synchronized product of all its factors. This is another transition system $\Theta_\mathcal{T} = \Theta_1 \otimes \dots \otimes \Theta_n = \langle S_\mathcal{T}, L, c, T_\mathcal{T}, s_\mathcal{T}^I, S_\mathcal{T}^G \rangle$, where $S_\mathcal{T} = S_1 \times \dots \times S_n$ is the Cartesian product of each factor's states, $T_\mathcal{T} = \{(\langle s_1, \dots, s_n \rangle, \ell, \langle t_1, \dots, t_n \rangle) \mid (s_i, \ell, t_i) \in T_i \text{ for } i \in \{1, \dots, n\}\}$, $s_\mathcal{T}^I = \langle s_1^I, \dots, s_n^I \rangle$, and $S_\mathcal{T}^G = S_1^G \times \dots \times S_n^G$. Note that the order of the factors is fixed, but arbitrary, as the resulting products for different orderings are all isomorphic. Throughout the paper, we consistently use subscripts to differentiate states in $S_\mathcal{T}$ (e.g., s, s', t) and their corresponding facts Θ_i (e.g., s_i, s_i', t_i).

Figure 1 shows the task of our running example in which a truck with limited capacity of one package must deliver two packages from A to B. The task has 4 factors, $\langle v_t, v_{p^1}, v_{p^2}, v_c \rangle$. The factors of v_{p^1} and v_{p^2} are identical, except for switching "1" and "2" in all labels. v_c models that the truck must have capacity (C_1) to load a package. Unloading does not require C_0, as this is always the case when a package has been loaded.

We use the factored representation because it is convenient to describe dominance analysis methods, and more expressive than other standard representations, such as STRIPS [3] and SAS$^+$ [1].[1]

2.2 *Dominance Analysis*

A relation $\preceq$ on a transition system Θ is a set of pairs of states, i.e., $\preceq \subseteq S \times S$. The identity relation is denoted by $\preceq^{id} := \{(s, s) \mid s \in S\}$. A relation $\preceq$ is reflexive if $\preceq^{id} \subseteq \preceq$. Dominance analysis methods take as input a planning task $\mathcal{T}$ without zero-cost labels[2] and automatically derive a dominance relation for $\Theta_\mathcal{T}$.

Definition 2. *A relation $\preceq$ is a dominance relation for a TS Θ if $s \preceq t$ implies $h^*(t) \leq h^*(s)$ for all $s, t \in S$.*

To prove that a relation is a dominance relation, one typically shows that it is a goal-respecting cost-simulation. A relation is goal-respecting if whenever $s \preceq t$, $s \in S^G$ implies that $t \in S^G$. A relation is a cost-simulation if whenever $s \preceq t$, for every transition $s \xrightarrow{\ell} s'$, there exists a transition $t \xrightarrow{\ell'} t'$ such that $s' \preceq t'$ and $c(\ell') \leq c(\ell)$. We always consider a *noop* transformation, adding a 0-cost label *noop* with a transition $s \xrightarrow{noop} s$ for all $s \in S$. This allows to reply to any transition $s \xrightarrow{\ell} s'$ with $t \xrightarrow{noop} t$ if $s' \preceq t$.

In order to represent a relation over the exponentially many states in the planning task, we use a compositional representation based on having a tuple of fact relations $\langle \preceq_1, \dots, \preceq_n \rangle$, one for each factor $\preceq_i \subseteq S_i \times S_i$. A fact relation $\preceq_i$ is called admissible when, for all $s, t \in S$ with $s_i \preceq_i t_i$ and $s_j = t_j$ for every $j \neq i$, we have $h^*(t) \leq h^*(s)$. Then, we define the product $\bigotimes\langle \preceq_1, \dots, \preceq_n \rangle := \{(s, t) \mid \forall i \in \{1, \dots, n\} : s_i \preceq_i t_i\}$. The resulting $\bigotimes\langle \preceq_1, \dots, \preceq_n \rangle$ is a dominance relation if all $\preceq_i$ are admissible.

Although deciding whether a fact relation is admissible is hard, we can compute informative admissible relations in polynomial time in the size of the task using *label-dominance simulations*.

Definition 3. *Let Θ_j be a TS, $\preceq_j \subseteq S_j \times S_j$, and $\ell, \ell' \in L$. ℓ' dominates ℓ in Θ_j given $\preceq_j$ if $\forall s_j \xrightarrow{\ell} s_j' : \exists s_j \xrightarrow{\ell'} t_j' : s_j' \preceq_j t_j'$. Given a factored relation $\langle \preceq_1, \dots, \preceq_n \rangle$ for a task $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n \rangle$, we define $\sqsubseteq_i := \{(\ell, \ell') \mid c(\ell') \leq c(\ell) \wedge \forall j \in \{1, \dots, n\}, j \neq i : \ell'$ dominates ℓ in Θ_j given $\preceq_j\}$.*

Intuitively, the meaning is that ℓ' is at least as good as ℓ in all remaining factors, so that when we compare transitions on Θ_i, a transition with ℓ' can be used to reply to a transition with ℓ.

Definition 4. *Let $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n \rangle$ be a factored task. A tuple $\langle \preceq_1, \dots, \preceq_n \rangle$ of relations $\preceq_i \subseteq S_i \times S_i$ is a label-dominance simulation for $\mathcal{T}$ if all $\preceq_i$ are goal-respecting and whenever $s_i \preceq_i t_i$:*

$$\forall s_i \xrightarrow{\ell} s_i' : \exists t_i \xrightarrow{\ell'} t_i' : s_i' \preceq_i t_i' \wedge \ell \sqsubseteq_i \ell'$$

The maximal relations satisfying the conditions of a label-dominance simulation are denoted by $\preceq^{\mathrm{LD}} = \langle \preceq_1^{\mathrm{LD}}, \dots, \preceq_n^{\mathrm{LD}} \rangle$. Such a maximal label-dominance simulation always exists and can be computed in polynomial time by a fix-point algorithm [17]. Consider for a moment a variant of the example of Figure 1 without the capacity factor (i.e., the truck can fit all packages). In that case, $\preceq^{\mathrm{LD}}$ is $\{(T_A, T_A), (T_B, T_B)\}$ for v_t, and $\{(P_A^i, P_A^i), (P_A^i, P_T^i), (P_A^i, P_B^i), (P_T^i, P_T^i), (P_T^i, P_B^i), (P_B^i, P_B^i)\}$ for the packages v_{p^i}. For example, (P_T^1, P_B^1), because for any transition with $unload(1, *)$ from P_T^1, there exists a transition $P_B^1 \xrightarrow{noop} P_B^1$ such that $P_*^1 \preceq P_B^1$ and *noop* dominates $unload(1, *)$ in all other factors (v_t and v_{p^2}).

3 Conditional Dominance

While considering fact relations is key for an efficient representation of dominance relations, requiring them to be admissible imposes a severe limitation on the dominance relations that can be obtained. Specifically, whenever a fact dominates another ($s_i \preceq_i t_i$), we require that t_i is at least as good as s_i in all circumstances. Here, we are interested in restricting the set of circumstances under which we consider t_i to be at least as good as s_i. For example, T_B is not always at least as good as T_A, but it is when no package remains at A.

[1] STRIPS/SAS$^+$ can be compiled in polynomial time into a factored task with equal state space, but not vice versa [2].

[2] We assume $c(\ell) > 0$ to simplify the definition of dominance relation, though extending our work to support zero-cost actions is straightforward.

In our example with capacity (v_c), $\preceq^{\text{LD}}$ is $\{(P_A^i, P_A^i), (P_T^i, P_T^i), (P_B^i, P_B^i)\}$, $\{(T_A, T_A), (T_B, T_B)\}$, and $\{(C_0, C_0), (C_1, C_1), (C_0, C_1)\}$. That is, the only thing discovered beyond the identity relation is that having capacity (C_1) is always good. However, that does not lead to any pruning because no two states during the search can differ only in the truck's capacity, as larger capacity implies that a package is not in the truck. We would like to find out that having a package at the destination is at least as good as anywhere else ($\{(P_A^i, P_B^i), (P_T^i, P_B^i)\}$), as happened without the capacity constraint. This is not the case in $\preceq^{\text{LD}}$ because $P_T^i \xrightarrow{unload(i,B)} P_B^i$ cannot be simulated by $P_B^i \xrightarrow{noop} P_B^i$, as $unload(i, B)$ has a positive side effect on the capacity and $noop$ does not. Indeed, this would lead to an inadmissible fact relation, where $s = \langle T_B, P_T^1, P_A^2, C_0\rangle \preceq \langle T_B, P_B^1, P_A^2, C_0\rangle = t$ but the truck cannot perform any load/unload actions starting from t, so $h^*(t) = \infty > h^*(s) = 4$.[3]

3.1 Compositional Relations Under Conditions

As a starting point, we take a factored task $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n\rangle$ and a label-dominance simulation $\langle \preceq_1^{\text{LD}}, \dots, \preceq_n^{\text{LD}}\rangle$. Then, we choose one factor i and attempt to find a new relation on the facts of that factor $\preceq_i \subseteq S_i \times S_i$ such that $\preceq_i \not\subseteq \preceq_i^{\text{LD}}$. In other words, $\preceq_i$ contains a pair of facts $s_i \preceq_i t_i$ such that t_i is not at least as good as s_i in all circumstances, but only under certain *conditions* on the other factors. We assume, without loss of generality that $i = 1$. This simplifies the notation: we find a relation for Θ_1 and conditions refer to factors $\Theta_2 \dots \Theta_n$.

Definition 5. *Let $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n\rangle$ be a factored task. An external condition for Θ_1 is a tuple of non-empty relations $\langle C_2, \dots, C_n\rangle$ where $C_j \subseteq S_j \times S_j$ for all $j \in \{2, \dots, n\}$.*

In our example, to compute a new relation for v_{p^1}, we consider the conditions shown in Table 1. With C^Y, for example, we show that having the package at the goal location B is at least as good as having it at A whenever the truck has full capacity, regardless of the position of the truck and other packages (provided they are not in the truck).

The condition derived from the label-dominance simulation is denoted by $C^{\text{LD}} := \langle \preceq_2^{\text{LD}}, \dots, \preceq_n^{\text{LD}}\rangle$ and the identity condition is denoted by $C^{id} := \langle \preceq_2^{id}, \dots, \preceq_n^{id}\rangle$. Note that each condition refers to all factors except one, and for each factor we can select any subset of pairs of facts, so we have exponentially many possible conditions in the number of factors as well as their size. For now, we assume that we are given a predefined set of external conditions as input. Section 4 introduces strategies to automatically choose conditions.

Given a factor Θ_1, an external condition C, and a relation $\preceq_1^C$ on states of Θ_1, we can construct a relation for the entire state space.

Definition 6. *Let $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n\rangle$ be a factored task, let $C = \langle C_2, \dots, C_n\rangle$ be an external condition for Θ_1, and let $\preceq_1^C \subseteq S_1 \times S_1$. Then, the* conditional composition *of $\preceq_1^C$ and C is $\preceq^C := \bigotimes\langle \preceq_1^C, C_2, \dots, C_n\rangle$.*

Then, we can identify when such a composition results in a dominance relation.

Definition 7. *Let $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n\rangle$ be a factored task. A relation $\preceq_1^C \subseteq S_1 \times S_1$ is admissible for $\mathcal{T}$ under a condition C if for each $s, t \in S_{\mathcal{T}}$ if $(s_1, t_1) \in \preceq_1^C$ and $(s_j, t_j) \in C_j$ for all $j \in \{2, \dots, n\}$, then $h^*(t) \leq h^*(s)$.*

[3] In this case, t is unreachable suggesting that we could focus on conditions that approximate the set of reachable states. However, our framework allows arbitrary conditions, so there is no direct relation with mutexes.

	v_{p^2}	v_c	$\preceq_1^C$ (v_{p^1})
$C^X = \langle$	$\{(P_A^2, P_A^2), (P_B^2, P_B^2)\}$,	$\{(C_0, C_1)\}\rangle$	$\{(P_T^1, P_B^1)\}$
$C^Y = \langle$	$\{(P_A^2, P_A^2), (P_B^2, P_B^2)\}$,	$\{(C_1, C_1)\}\rangle$	$\{(P_A^1, P_B^1)\}$
$C^Z = \langle$	$\{(P_T^2, P_T^2)\}$,	$\{(C_0, C_0)\}\rangle$	$\{(P_A^1, P_B^1)\}$

Table 1. Three conditions for our running example and conditional dominance relation for v_{p^1}. The factor v_t is omitted as it is the identity relation ($\{(T_A,T_A), (T_B,T_B)\}$) for all conditions.

In other words, $\preceq_1^C$ is admissible for $\mathcal{T}$ under a condition C if and only if the composition $\preceq^C$ is a dominance relation for $\Theta_{\mathcal{T}}$. This generalizes the previous notion of admissible fact relation, which we get if we choose the identity condition C^{id}. Previous (unconditional) dominance analysis methods always assumed that all fact relations needed to be compositional with such an identity condition because "every fact is as good as itself". Indeed, it holds that label-dominance simulations are always reflexive ($\preceq^{id} \subseteq \preceq^{\text{LD}}$). This assumption has some advantages as shown by the following theorem.

Theorem 1. *Let $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n\rangle$ be a factored task, and $R = \langle \preceq_1, \dots, \preceq_n\rangle$ a relation for each factor. If for all $i \in \{1, \dots, n\}$, $\preceq_i$ is admissible for $\mathcal{T}$ under $\preceq^{id}$, then for all $i \in \{1, \dots, n\}$ $\preceq_i$ is admissible for $\mathcal{T}$ under $\langle \preceq_1, \dots, \preceq_{i-1}, \preceq_{i+1}, \dots, \preceq_n\rangle$.*

Proof. As all $\preceq_i$ are admissible for $\mathcal{T}$ under $\preceq^{id}$, their composition is a dominance relation [17], which in turn implies admissibility under R. $\square$

That is the reason why label-dominance simulation methods compute relations for all factors simultaneously. However, this is no longer the case for the general case of conditional dominance when some of the relations are not reflexive.

3.2 Conditional Label-Dominance Simulation

We now introduce our approach to compute additional state dominances based on external conditions. We are given a factored task $\mathcal{T}$, a label-dominance simulation $\preceq^{\text{LD}}$, and a predefined set of conditions $\mathcal{C} = \{C^1, \dots, C^k\}$ for Θ_1, where $C^i = \langle C_2^i, \dots, C_n^i\rangle$. Whenever the identifier of the condition is not relevant, we simply write $C \in \mathcal{C}$ to refer to some condition in the set.

As with the standard label-dominance simulation method, we compute a relation $\preceq_1^C$ for Θ_1 for each $C \in \mathcal{C}$ such that for each $s_1 \preceq_1^C t_1$ and transition $s_1 \xrightarrow{\ell} s_1'$, t_1 has a response $t_1 \xrightarrow{\ell'} t_1'$ where t_1' "is at least as good as" s_1' and ℓ' "is at least as good as" ℓ. There are two key changes in how transitions from s_1 and t_1 are compared. On the one hand, we can ignore all transitions $s_1 \xrightarrow{\ell} s_1'$ such that ℓ is not applicable in any state consistent with the current context established by C. The set of relevant labels is defined as $L(C) = \{\ell \mid \forall i \in \{2, \dots, n\} : \exists s_i, s_i', t_i \in S_i : s_i \xrightarrow{\ell} s_i' \wedge (s_i, t_i) \in C_i\}$. In other words, a label is relevant under a condition if, for every factor, there exists at least one fact consistent with the condition where the label is applicable. Note that considering this was unnecessary in label-dominance simulation, as $L(C) = L$ if all C_i are reflexive.

On the other hand, we change how to compare transitions (i.e., formally define "at least as good" to determine if $s_1 \xrightarrow{\ell} s_1'$ can be simulated by $t_1 \xrightarrow{\ell'} t_1'$). This depends on two conditions (C^{bef} and C^{aft}), used to compare the states before and after applying the transition. $C^{bef} \in \mathcal{C}$ is the condition that restricts the states before applying the transition (as we are computing whether $s_1 \preceq^{bef} t_1$). $C^{aft} \in \mathcal{C} \cup \{C^{\text{LD}}\}$ is used to compare the states after the transition to determine if t_1' is at least as good as s_1'. So, when comparing the labels ℓ and ℓ' we must ensure that, if we start in any two states s,t where C^{bef} holds (i.e. $(s_i, t_i) \in C_i$ for all $i \in \{2, \dots, n\}$), and apply any ℓ-labelled transition on s, we can choose an ℓ'-labelled transition on t such that condition C^{aft} necessarily holds for the resulting states s' and t'. As conditions are factored, this is tested independently for each factor:

Definition 8 (Conditional Label-Dominance)**.** *Let Θ_i be a factor, let $\preceq^{bef}, \preceq^{aft} \subseteq S_i \times S_i$, and let $\ell, \ell' \in L$. Then, ℓ'* conditionally dominates ℓ in Θ_i with respect to $\preceq^{bef}$ and $\preceq^{aft}$ *if and only if*

$$\forall s \preceq^{bef} t : \forall s \xrightarrow{\ell} s' : \exists t \xrightarrow{\ell'} t' : \left(s' \preceq^{aft} t'\right)$$

For each pair of conditions $C^{bef}, C^{aft} \in \mathcal{C}$ we define a label relation $\sqsubseteq^{C^{aft}}_{C^{bef}} \subseteq L \times L$ such that $\ell \sqsubseteq^{C^{aft}}_{C^{bef}} \ell'$ if and only if $c(\ell') \leq c(\ell)$ and for all $i \in \{2, \dots, n\}$, ℓ' conditionally dominates ℓ in Θ_i with respect to C_i^{bef} and C_i^{aft}.

This generalizes the standard notion of label dominance (Def. 3), which we get by choosing $C^{bef} = C^{id}$, and $C^{aft} = C^{\mathrm{LD}}$. Intuitively C^{bef} represents under which situations ℓ' is at least as good as ℓ. If $C^{bef} = C^{id}$, we are considering whether ℓ' dominates ℓ (a) in all circumstances, and (b) when starting from the same place. Conditional label-dominance allows us to challenge (a) and ask whether ℓ' dominates ℓ in some specific circumstances. For example, by choosing $C^{bef} \subset C^{id}$, the analysis of whether ℓ' is at least as good as ℓ is restricted only to certain situations. Thus, only transitions with ℓ that are possible in all factors under the current context C^{bef} need to be considered. The condition C^Y in our running example, has a single pair (C_1, C_1) for v_c, so only states where the truck capacity is available are considered.

We can also challenge (b) with some $C^{bef} \not\subseteq C^{id}$ (i.e., where $(s_i, t_i) \in C^{bef}$ for some $s_i \neq t_i$). When we do that, we are able to say that ℓ' conditionally dominates ℓ when we consider if t_i dominates s_i. Consider for example, the condition C^X from Table 1 containing the pair $(C_0$,$C_1)$. This allows us to express that $unload(1, B) \sqsubseteq^{C^{\mathrm{LD}}}_{C^X} noop$, i.e., $noop$ conditionally dominates $unload(1, B)$ in the context of C^X because the truck capacity is already available when we apply $noop$.

Finally, C^{aft} is used to compare the resulting states after applying ℓ and ℓ'. Normally, we should set this to C^{LD}, i.e., states that we have already proven to be at least as good in every circumstance. However, sometimes, we want to be more specific. When we use $C^{aft} \neq C^{\mathrm{LD}}$, we can guarantee that by applying ℓ' we can reach some state for which C^{aft} holds in all other factors. For example, we can say that the transition $P^1_T \xrightarrow{unload(1,A)} P^1_A$ is simulated by $P^1_B \xrightarrow{noop} P^1_B$ under C^X because P^1_A is dominated by P^1_B under C^Y and after applying those transitions C^Y is guaranteed to hold.

One needs to be careful because $\sqsubseteq^{C^{aft}}_{C^{bef}}$ is no longer always reflexive if C^{bef} or C^{aft} are not reflexive. That is, to show that t_1 dominates s_1 under a certain condition C we cannot always reply to some transition $s_1 \xrightarrow{\ell} s_1$ with the same transition $t_1 \xrightarrow{\ell} t_1$. This may be counter-intuitive as in traditional simulation relations a self-loop can always be answered by another self-loop with the same label. But there is a good reason for this: if for example transition $s_1 \xrightarrow{\ell'} s_1$ was ignored due to ℓ' not being applicable in context C, then it would be problematic because s_1 could have an applicable sequence $s_1 \xrightarrow{\ell} s_1 \xrightarrow{\ell'} s_1$ for which t_1 does not have a response. This can happen if after applying the transition ℓ, we can reach some state in which the condition C does not hold for some other factor. It is important to observe that this does not apply to $noop$, and it is always the case that $noop \sqsubseteq^C_C noop$ for any $C \in \mathcal{C}$. This means that any transition $s_1 \xrightarrow{noop} s_1$ can be simulated by $t_1 \xrightarrow{noop} t_1$. Therefore, as in previous work, adding $noop$ transitions is never detrimental and in practice any transition $s_1 \xrightarrow{noop} s_1$ can be ignored.

Finally, while LD simulations require that all relations are goal-respecting, this is only necessary for *goal-relevant* conditions.

Definition 9. *A relation $\preceq$ is* goal-relevant *if $\exists s \preceq t : s \in S^G$. A condition $C = \langle C_2, \dots, C_n \rangle$ is goal-relevant (goal-respecting) if all $C_j \in C$ are goal-relevant (goal-respecting).*

Any reflexive relation (as the ones in LD simulation) is always goal-relevant, but now we can use conditions that are not goal-relevant so that we are not required to always be goal-respecting.

With this, we have all the ingredients we need to introduce our notion of conditional label-dominance simulation.

Definition 10 (Conditional LDS)**.** *Let $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n \rangle$ be a factored task, $\langle \preceq^{LD}_1, \dots, \preceq^{LD}_n \rangle$ be a label-dominance simulation for $\mathcal{T}$, and $\mathcal{C} = \{C^1, \dots, C^k\}$ be a set of external conditions for Θ_1 where $C^i = \langle C^i_2, \dots, C^i_n \rangle$. Then, a set of relations $\{\preceq^{C^1}_1, \dots, \preceq^{C^k}_1\}$ with $\preceq^{C^i}_1 \subseteq S_1 \times S_1$ for all $i \in \{1, \dots, k\}$ is a* conditional label-dominance simulation (CLDS) for Θ_1 under $\mathcal{C}$ *if for all $i \in \{1, \dots, k\}$ (1) if C^i and $\preceq^{C^i}_1$ are goal-relevant, then C^i and $\preceq^{C^i}_1$ are goal-respecting, and (2) whenever $s_1 \preceq^{C^i}_1 t_1$:*

$$\forall s_1 \xrightarrow{\ell} s'_1 \text{ s.t. } \ell \in L(C^i) : \exists t_1 \xrightarrow{\ell'} t'_1 : \exists C^{aft} \in \mathcal{C} \cup \{C^{LD}\} :$$
$$s'_1 \preceq^{C^{aft}}_1 t'_1 \wedge (\ell, \ell') \in \sqsubseteq^{C^{aft}}_{C^i}$$

To obtain a dominance relation, we simply compute the union of all relations as per Def. 6, $\preceq^{\mathrm{LD}} \cup \bigcup_{C^i \in \mathcal{C}} \preceq^{C^i}$.

The equation is similar to that of label-dominance simulation (Def. 4) in that, for every possible transition that s_1 can make to s'_1, t_1 must be able to reply reaching some t'_1 that dominates s'_1. However, here the notion of label-dominance depends on the condition; s'_1 and t'_1 are compared in the context of another condition; only transitions with relevant labels are considered from s; and $\preceq^{C^j}_1$ is only required to be goal-respecting if C^j is goal-relevant.

In the example of Table 1, the relations in the right hand-side satisfy Def. 10. But crucially, all three conditions depend on each other, and cannot be proven individually. For example, in C^X, the transition $P^1_T \xrightarrow{unload(1,A)} P^1_A$ is simulated by $P^1_B \xrightarrow{noop} P^1_B$, relying on the fact that $P^1_A \preceq^{C^Y}_1 P^1_B$ and $unload(1, A) \sqsubseteq^{C^Y}_{C^X} noop$. This is not a problem, as we can still prove that the union of all these relations is a valid dominance relation for the planning task.

Theorem 2. *Let $\mathcal{T}$ be a factored task with a conditional LD simulation for Θ_1 under $\{C^1, \dots, C^k\}$. Then, $\preceq^{LD} \cup \bigcup_{C^i \in \mathcal{C}} \preceq^{C^i}$ is a goal-respecting cost-simulation for $\Theta_{\mathcal{T}}$.*

Proof. Let $s = (s_1, \dots, s_n), t = (t_1, \dots, t_n) \in S$ and $i \in \{1, \dots, k\}$ such that $(s, t) \in \preceq^{C^i}$.

First, we show that it is goal-respecting ($s \in S^G \implies t \in S^G$). If $s \in S^G$, then $s_j \in S^G_j$. As C^i and $\preceq^{C^i}_1$ are goal-relevant, so by Def. 10 both C^i and $\preceq^{C^i}_1$ are goal-respecting. Therefore, $t_j \in S^G_j$ for all $j \in \{1, \dots, n\}$, so $t \in S^G$.

To show that it is a cost-simulation, let $(s_1, \dots, s_n) \xrightarrow{\ell} (s'_1, \dots, s'_n)$ be any transition from s. By the definition of the synchronized product, $s_j \xrightarrow{\ell} s'_j$ for all $j \in \{1, \dots, n\}$. As $s_1 \preceq^{C^i}_1 t_1$, there exists a transition $t_1 \xrightarrow{\ell'} t'_1$ that satisfies the equation of Definition 10. Since $\ell \sqsubseteq^{C^{aft}}_{C^i} \ell'$, there also exists a transition $t_j \xrightarrow{\ell'} t'_j$ for every $j \in \{2, \dots, n\}$ such that $(s'_j, t'_j) \in C^{aft}_j$. Therefore, the pair

Algorithm 1: Conditional LDS

```
Input: Factored Task $\mathcal{T} = \langle \Theta_1, \dots, \Theta_n \rangle$
Input: $\preceq^{\mathrm{LD}}$: label-dominance simulation for $\mathcal{T}$
Input: $\mathcal{C}$: external conditions for $\Theta_1$
Output: $\preceq_1^{\mathcal{C}} = \{\preceq_1^C \mid C \in \mathcal{C}\}$: maximal cond. LDS
1  foreach $C \in \mathcal{C}$ do
2      if $C$ is goal-relevant and goal-respecting then
3          $\preceq_1^C \leftarrow \{(s_1, t_1) \mid s_1, t_1 \in S_1, s_1 \notin S_1^G \vee t_1 \in S_1^G\}$
4      else if $C$ is goal-relevant but not goal-respecting then
5          $\preceq_1^C \leftarrow \{(s_1, t_1) \mid s_1, t_1 \in S_1, s_1 \notin S_1^G\}$
6      else
7          $\preceq_1^C \leftarrow \{(s_1, t_1) \mid s_1, t_1 \in S_1\}$
8  while $\exists C \in \mathcal{C}$, $s_1, t_1 \in S_1$ s.t. $s_1 \preceq_1^C t_1$ and the equation of
       Definition 10 does not hold under $\preceq^{LD}$, $\preceq_1^{\mathcal{C}}$, and $\mathcal{C}$ do
9      remove $(s_1, t_1)$ from $\preceq_1^C$
10 return $\{\preceq_1^C \mid C \in \mathcal{C}\}$;
```

$\big((s'_1, \dots, s'_n), (t'_1, \dots, t'_n)\big)$ is contained in $\preceq^{C^{aft}}$ and consequently in the union $\preceq^{\mathrm{LD}} \cup \bigcup_{C^i \in \mathcal{C}} \preceq^{C^i}$. □

We call $\preceq^{\mathcal{C}} = \{\preceq_1^{C^1}, \dots, \preceq_1^{C^k}\}$ the *maximal conditional label-dominance simulation* for Θ_1 under the set of conditions $\mathcal{C} = \{C^1, \dots, C^k\}$ if for all other conditional label-dominance simulations $\{\preceq_1'^{C^1}, \dots, \preceq_1'^{C^k}\}$ for Θ_1 under $\mathcal{C}$, it is the case that $\preceq_1'^C \subseteq \preceq_1^C$ for all $C \in \mathcal{C}$.

Theorem 3. *There exist families of planning tasks under which pruning is possible with conditional dominance using polynomially many conditions, and no pruning is possible with label-dominance simulation under polynomial merge transformations.*

Proof Sketch. Our running example, scaling the number of packages, is such an example. Using conditional dominance it suffices to add an additional condition per package similar to C^Z but with a different package in the truck each time. However, this depends on the position of all packages so to achieve a similar relation with label-dominance simulation, one needs to consider the product of all the packages and the capacity of the truck, which grows exponentially with the number of packages. □

3.3 Computing Conditional LD Simulations

Algorithm 1 shows how to compute the maximal conditional label-dominance simulation in polynomial time. The procedure resembles how label-dominance simulations are computed: initializing all relations to an over-approximation and iteratively removing any pair that does not satisfy the conditions established in Definition 10.

Theorem 4. *For a given condition set $\mathcal{C}$, a* maximal conditional label-dominance simulation *always exists and can be computed in polynomial time in the size of $\mathcal{T}$, and $\mathcal{C}$.*

Proof sketch. A unique maximal CLDS exists, because if any two sets of relations satisfy Def. 10, then their union does too. For (1), we distinguish three cases. If C^i is not goal-relevant, then (1) holds trivially. If C^i is goal-relevant but not goal respecting, then both $\preceq_1^{C^i}$ and $\preceq_1'^{C^i}$ must also be not goal-relevant, and their union is therefore not goal-relevant. If C^i is goal-relevant and goal-respecting, then both $\preceq_1^{C^i}$ and $\preceq_1'^{C^i}$ must be goal-respecting (since any non-goal-relevant relation is always goal-respecting), and so their union is goal-respecting as well.

For condition (2), enlarging the relations by adding pairs cannot invalidate pairs that already satisfy the condition. Thus, if two sets of relations satisfy the conditions, the set of their unions does too.

Algorithm 1 always terminates as a pair (s_1, t_1) is removed in each iteration and there are finitely many of them. The final result satisfies the condition of Definition 10 (as the empty relation always does). The result is maximal because removing pairs from each $\preceq_1^{C^i}$ can only cause other pairs to be removed, and each $\preceq_1^C$ is initialized with an over approximation, so any pair removed cannot be part of any conditional label-dominance simulation. □

4 Finding Sets of External Conditions

Finding external conditions is a complex problem, as the space of possible conditions is exponential both in the number of factors as well as in the size of those factors. Sampling such a space randomly is hopeless, as most conditions result in empty dominance relations. Therefore it is important to guide the search of conditions by considering the impact of including/excluding a given fact pair on the conditional label dominance. However, this creates conflicting objectives. As with standard label-dominance simulations, we prefer to have as many pairs as possible in each condition C because (a) that makes C hold in more states therefore maximizing the size of the resulting dominance relation and (b) the more pairs are included in $\preceq^{aft}$, the more label dominances we can find. However, we also need to keep conditions $\preceq^{bef}$ small to find more label dominances and reduce the set of relevant labels. As those objectives are conflicting, there is no monotonicity property to apply a fix-point algorithm in the same way as we had for label-dominance simulations.

We introduce three strategies to compute conditions. They make several simplifying assumptions that bias the type of conditions that can be found. We focus on generating a single condition for every factor. While Definition 10 benefits from computing relations for multiple conditions at the same time (as shown by our running example), we leave the exploration of that idea for future work. The main idea is to compute the external condition simultaneously with $\preceq_1^C$, making greedy commitments in order to ensure termination.

4.1 Conditions Underapproximating LDS

Algorithm 2 shows our first approach, which computes the external condition within the loop of Algorithm 1. The set of external conditions $\mathcal{C}$ is initialized with a single condition, C, equal to the label-dominance simulation. The main modification with respect to Algorithm 1 is that if a pair (s_1, t_1) does not fulfill the condition of Def 10, we attempt to remove fact pairs from C instead of removing the pair. The algorithm also keeps a list of greedily committed label dominances. Whenever it was necessary that $\ell \sqsubseteq_C^C \ell'$ in order to keep $(s_1, t_1) \in \preceq_1^C$, we store the pair (ℓ, ℓ'). Any change to C negatively affecting these dominances is forbidden. Note that the same is not necessary for $\sqsubseteq_C^{C^{\mathrm{LD}}}$. As the LD simulation is fixed, $s'_1 \preceq_1^{\mathrm{LD}} t'_1$ will always hold, and all conditional label dominances in $\sqsubseteq_C^{C^{\mathrm{LD}}}$ will always be preserved because C can only get smaller and C^{LD} remains unchanged. To reduce the amount of commitments, we consider four possibilities to simulate each transition $s_1 \xrightarrow{\ell} s'_1$ in order:

1. Use $\preceq^{\mathrm{LD}}$ (line 8), which does not require any commitment.
2. Use $\preceq_1^C$ (line 11), committing to preserve $\ell \sqsubseteq_C^{C^{\mathrm{LD}}} \ell'$.
3. Modify the condition C. First, "almost dominances" are identified, i.e., label pairs (ℓ, ℓ') such that if ℓ' would dominate ℓ then t_1 could simulate $s_1 \xrightarrow{\ell} s'_1$ and the condition can be changed by

Algorithm 2: Conditions Underapproximating LDS

```
   Input: $\mathcal{T}$: factored task
   Input: $\preceq^{LD}$: label-dominance simulation for $\mathcal{T}$
   Output: External condition for $\Theta_1$
1  $C \leftarrow \langle \preceq_2^{LD}, \dots, \preceq_n^{LD} \rangle$;
2  $label\_dominances \leftarrow \{\}$ ;
3  $\preceq_1^C \leftarrow \{(s_1, t_1) \mid s_1 \notin S_1^G \lor t_1 \in S_1^G\}$;
4  while $\exists (s_1, t_1) \in \preceq_1^C$ and the equation of Definition 10 does
     not hold under $\preceq^{LD}$, $\preceq_1^C$, and $\mathcal{C}$ do
5      select one such $(s_1, t_1)$;
6      Save $C$ and $label\_dominances$ ;
7      forall $s_1 \xrightarrow{\ell} s_1'$ such that $\ell \in L(C)$ do
8          if $\exists t_1 \xrightarrow{\ell'} t_1'.\ s_1' \preceq_1^{LD} t_1' \land \ell \sqsubseteq_C^{C^{LD}} \ell'$ then
9              continue;
10         if $\exists t_1 \xrightarrow{\ell'} t_1'.\ (s_1', t_1') \in \preceq_1^C \land \ell \sqsubseteq_C^C \ell'$ then
11             select one such transition $t_1 \xrightarrow{\ell'} t_1'$ preferring
                 those such that $(\ell, \ell') \in label\_dominances$ ;
12             add $(\ell, \ell')$ to $label\_dominances$;
13             continue;
14         if $\exists (t_1 \xrightarrow{\ell'} t_1').\ (s_1', t_1') \in \preceq_1^C$
             $\land AlmostDom(\ell', \ell, \preceq^{LD}, label\_dominances)$ then
15             remove pairs from $C_k$ and $C_{k'}$ until $\ell \sqsubseteq_C^{C^{\mathrm{LD}}} \ell'$;
16             continue;
17         if $\exists (t_1 \xrightarrow{\ell'} t_1').\ (s_1', t_1') \in \preceq_1^C$
             $\land AlmostDom(\ell', \ell, C, label\_dominances)$ then
18             remove pairs from $C_k$ and $C_{k'}$ until $\ell \sqsubseteq_C^C \ell'$;
19             add $(\ell, \ell')$ to $label\_dominances$;
20             continue;
21         Remove $(s_1, t_1)$ from $\preceq_1^C$;
22         Restore $C$ and $label\_dominances$ (as per line 6);
23         break;
24 return $C$, $\preceq_1^C$;
```

affecting at most two factors C_k and $C_{k'}$, without breaking any of the previously committed label dominances. Then, we remove from C_k and $C_{k'}$ any pairs (s_k, t_k) or $(s_{k'}, t_{k'})$ failing the test of Definition 8. We first use $\preceq^{\mathrm{LD}}$ to avoid commitments.

4. Same as 3., but using $\preceq_1^C$.

If this process fails, we give up on the pair s_1, t_1, and restore C and the committed label dominances to their previous state. The algorithm is guaranteed to terminate for the same reasons as Algorithm 1. At the end of the process, $\preceq_1^C$ is guaranteed to be a conditional label-dominance simulation for the returned C. A problem with this approach is that it makes a lot of greedy commitments, and we lack good "heuristics" to guide the process (i.e., decide what pairs s_1, t_1 to focus first). Therefore, the algorithm is often time consuming and can often fail, finishing with an empty relation.

4.2 *Conditions based on Fact Pairs*

Our second approach is parameterized by a fact pair $(s_1, t_1) \notin \preceq_1^{\mathrm{LD}}$ and attempts to find a condition C such that $s_1 \preceq_1^C t_1$. We run this from every such pair. The condition is initialized to the label-dominance simulation, but in this case $\preceq_1^C$ is initialized as a single pair (s_1, t_1) and more pairs are inserted on demand. For example, if $t_1 \xrightarrow{\ell} t_1'$ is the only possible response to $s_1 \xrightarrow{\ell} s_1'$, then (s_1', t_1') is added to $\preceq_1^C$. The algorithm iterates over all pairs added to $\preceq_1^C$ and, similarly to Algorithm 2, attempts to modify the condition C and/or add pairs to $\preceq_1^C$ in order to meet the definition. Every time that we insert a pair on $\preceq_1^C$, it is a commitment, so if the conditions of Definition 10 cannot be met for such a pair the algorithm ends in failure.

4.3 *Online Conditions based on State Pairs*

The previous strategies are agnostic to the states generated during the search. This causes that, even if some fact-dominance relations are found, they may be useless due to never encountering during the search a pair of states (s, t) that meets the condition.

Our third strategy, CLDS Online, is triggered during the search, when two states s, t are compared and t dominates s in all factors except one (here, we assume wlog Θ_1). The question is whether we can find an external condition C such that $s \preceq^C t$ to prune s. Therefore, any successful condition will at least be useful to prune a state. Of course, we still desire C to contain as many fact-pairs as possible so that it can be used to prune other states during the search.

By targeting a specific pair s, t, CLDS Online can initialize the condition of each factor with a single pair $C_i = \{(s_i, t_i)\}$ for $i \in \{2, \dots, n\}$. The algorithm then attempts to prove that under such a condition $s_1 \preceq_1^C t_1$. If such a check fails, due to a transition $s_1 \xrightarrow{\ell} s_1'$ not being simulated by any $t_1 \xrightarrow{\ell'} t_1'$, new fact pairs are added to C so that $\ell \sqsubseteq_C^{C^{\mathrm{LD}}} \ell'$ or $\ell \sqsubseteq_C^C \ell'$. Note that, at the beginning of the algorithm, C is non-reflexive so most labels do not even dominate themselves.

5 Empirical Evaluation

We implemented our conditional dominance approach in Fast Downward [7] with the existing implementation for computing label-dominance simulations [17]. As benchmark set we use the optimal-track instances from the International Planning Competitions from 1998 to 2018, ignoring domains with conditional effects. We ran experiments using Lab [10] with a time limit of 30 minutes and a memory limit of 4 GB. Source code and results are publicly available [20].

The main purpose of our experiment is to test whether conditional label-dominance simulation can yield more informative dominance relations than the standard LD simulation. To that end, we compare the amount of nodes expanded by uniform-cost search with dominance pruning. Whenever a state s is generated during the search we compare it to the initial state, its parent, and its siblings with lower or equal g-value. If any of them dominates s we prune it without inserting it into the open list. In principle, nodes could also be compared against all other previously seen states [17], but that would require new data structures and is left for future work.

Figure 2 shows that the three strategies are able to find useful conditions. Even with the restrictions imposed by our strategies for finding external conditions, and only comparing each state against a few other states, conditional label-dominance simulations can achieve pruning in several domains. The domain with most pruning is Blocksworld. This is a very good example to showcase the power of conditional dominance. As in our running example, label-dominance simulation methods do not find any useful pruning in that domain, even when considering sets of variables. In Blocksworld, however, a single condition suffices to show that having a block on the table is at least as good as on some non-goal block. Our strategies, specially CLDS Online, are able to capture this, heavily reducing the search space by up to two orders of magnitude.

The method's effectiveness depends heavily on the strategy for generating conditions. The online configuration generally performs best, as offline configurations lack knowledge of which states will

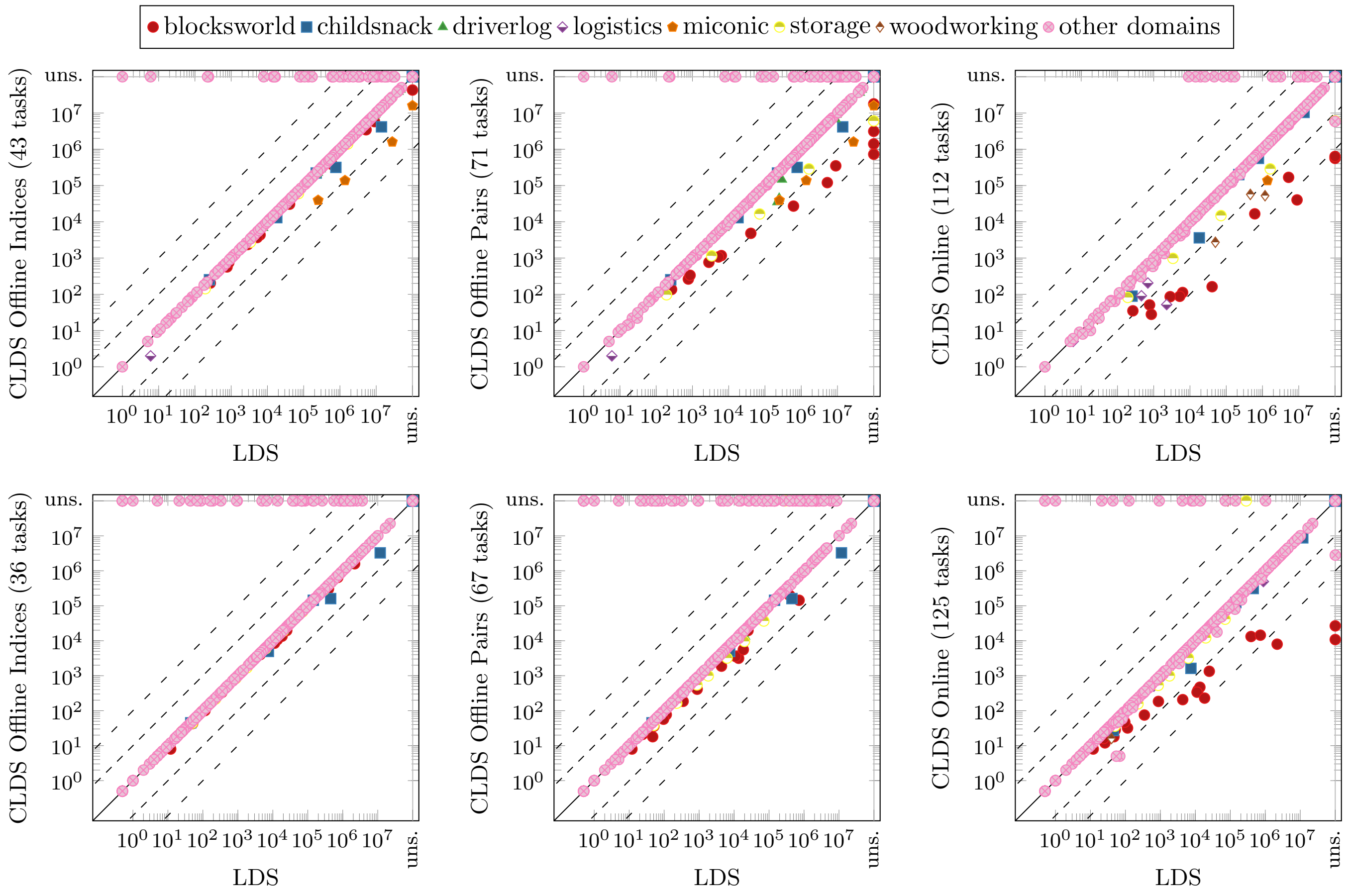

Figure 2. Expansions until last f-layer of uniform-cost search (above) and A* with LM-cut (below) using dominance pruning with LD-simulation versus conditional LD simulations. Every plot uses conditions derived with one of our three strategies: Offline indices (left), offline pairs (middle), and online (right). In parenthesis we indicate in how many tasks we achieve more pruning than the baseline.

be compared during the search. However, they can still be useful in certain domains, such as Miconic.

Our current implementation does not yet improve the efficiency of optimal planners. To test this, we applied dominance pruning with A* [6] using the LM-cut heuristic [8] (see the lower part of Figure 2). The number of tasks where conditional dominance has an impact is comparable to that of uniform-cost search. However, when using a heuristic the impact of pruning diminishes, as the heuristic already avoids the expansion of some of the states that would be pruned. The two offline configurations are most affected, as they do not take into consideration what kind of states are encountered during the search. Consequently, Blocksworld is the only domain where coverage increases and only when using the online configuration.

In many other domains the overhead of computing dominance decreases the performance significantly. While computing conditional dominance relations is not much more expensive than computing label-dominance simulations, the additional search for conditions and their respective conditional dominance relations only pays off when pruning yields a substantial reduction in expanded states. The overhead is largest in the online configuration, as it performs computation at each expansion during the search. However, the results in Blocksworld show that, in domains where dominance pruning is successful at significantly reducing the number of expanded states, the resulting savings can outweigh the overhead and increase coverage.

6 Conclusions

In this paper, we have introduced conditional dominance, as a new way of comparing states on any planning task. Previous methods looked for facts that are always at least as good as others in every situation. However, in some domains there is no such pair of facts, so they cannot derive useful dominance relations. Reasoning about multiple variables together can help, but the computational cost grows exponentially in the number of variables. In conditional dominance, we use a set of conditions, which identify relevant contexts under which it can be shown that certain facts are better than others. These contexts take into account all other variables of the task and find dominance relations in domains where the previous methods could not. A challenge is how to automatically derive useful conditions. We provided several approaches that can find relevant conditions automatically in many domains.

Our experiments serve as a proof of concept, demonstrating that conditional dominance can derive useful information beyond previous dominance analysis methods for classical planning. However, more research is required to effectively take advantage of conditional dominance to speed-up search. This opens many avenues for future work. The space of conditions is huge, and better methods are needed to systematically find good conditions. Also, the method could be combined with other extensions of dominance analysis, such as quantitative dominance [13] and contrastive analysis [16, 12].

Acknowledgements

This research was supported by the Independent Research Fund Denmark through a Sapere Aude: DFF-Starting Grant under reference number 3120-00063B.

References

[1] C. Bäckström and B. Nebel. Complexity results for SAS$^+$ planning. *Computational Intelligence*, 11(4):625–655, 1995.

[2] C. Büchner, P. Ferber, J. Seipp, and M. Helmert. Abstraction heuristics for factored tasks. In S. Bernardini and C. Muise, editors, *Proceedings of the Thirty-Fourth International Conference on Automated Planning and Scheduling (ICAPS 2024)*, pages 40–49. AAAI Press, 2024.

[3] T. Bylander. The computational complexity of propositional STRIPS planning. *Artificial Intelligence*, 69(1–2):165–204, 1994.

[4] J. Eisenhut, Á. Torralba, M. Christakis, and J. Hoffmann. Automatic metamorphic test oracles for action-policy testing. In S. Koenig, R. Stern, and M. Vallati, editors, *Proceedings of the Thirty-Third International Conference on Automated Planning and Scheduling (ICAPS 2023)*, pages 109–117. AAAI Press, 2023.

[5] M. Ghallab, D. Nau, and P. Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann, 2004.

[6] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[7] M. Helmert. The Fast Downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

[8] M. Helmert and C. Domshlak. Landmarks, critical paths and abstractions: What's the difference anyway? In A. Gerevini, A. Howe, A. Cesta, and I. Refanidis, editors, *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*, pages 162–169. AAAI Press, 2009.

[9] M. Helmert, P. Haslum, J. Hoffmann, and R. Nissim. Merge-and-shrink abstraction: A method for generating lower bounds in factored state spaces. *Journal of the ACM*, 61(3):16:1–63, 2014.

[10] J. Seipp, F. Pommerening, S. Sievers, and M. Helmert. Downward Lab. https://doi.org/10.5281/zenodo.790461, 2017.

[11] S. Sievers and M. Helmert. Merge-and-shrink: A compositional theory of transformations of factored transition systems. *Journal of Artificial Intelligence Research*, 71:781–883, 2021.

[12] R. G. Tollund, K. G. Larsen, and Á. Torralba. What makes you special? contrastive heuristics based on qualified dominance. In *Proceedings of the 34th International Joint Conference on Artificial Intelligence (IJCAI 2025)*. IJCAI, 2025.

[13] Á. Torralba. From qualitative to quantitative dominance pruning for optimal planning. In C. Sierra, editor, *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI 2017)*, pages 4426–4432. IJCAI, 2017.

[14] Á. Torralba. Completeness-preserving dominance techniques for satisficing planning. In J. Lang, editor, *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI 2018)*, pages 4844–4851. IJCAI, 2018.

[15] Á. Torralba. On the optimal efficiency of A* with dominance pruning. In K. Leyton-Brown and Mausam, editors, *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021)*, pages 12007–12014. AAAI Press, 2021.

[16] Á. Torralba. Reshaping state-space search: From dominance to contrastive analysis. In Y. Chen and J. Neville, editors, *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI 2023)*, page 15457. AAAI Press, 2023.

[17] Á. Torralba and J. Hoffmann. Simulation-based admissible dominance pruning. In Q. Yang and M. Wooldridge, editors, *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*, pages 1689–1695. AAAI Press, 2015.

[18] Á. Torralba and P. Kissmann. Focusing on what really matters: Irrelevance pruning in merge-and-shrink. In L. Lelis and R. Stern, editors, *Proceedings of the Eighth Annual Symposium on Combinatorial Search (SoCS 2015)*, pages 122–130. AAAI Press, 2015.

[19] Á. Torralba and S. Sievers. Merge-and-shrink task reformulation for classical planning. In S. Kraus, editor, *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI 2019)*, pages 5644–5652. IJCAI, 2019.

[20] A. Wilhelm and Á. Torralba. Code and experiment data for the ECAI 2025 paper "Conditional Dominance Analysis for Classical Planning". https://doi.org/10.5281/zenodo.16938058, 2025.