Stubborn Sets for Fully Observable Nondeterministic Planning

Dominik Winterer and Robert Mattmüller
University of Freiburg, Germany
{wintered, mattmuel}@informatik.uni-freiburg.de

Martin Wehrle
University of Basel, Switzerland
martin.wehrle@unibas.ch

Abstract

The stubborn set method is a state-space reduction technique, originally introduced in model checking and then transferred to classical planning. It was shown that stubborn sets significantly improve the performance of optimal deterministic planners by considering only a subset of applicable operators in a state. Fully observable nondeterministic planning (FOND) extends the formalism of classical planning by nondeterministic operators. We show that stubborn sets are also beneficial for FOND problems. We introduce nondeterministic stubborn sets, stubborn sets which preserve strong cyclic plans. We follow two approaches: Fast Incremental Planning with stubborn sets from classical planning and LAO* search with nondeterministic stubborn sets. Our experiments show that both approaches increase coverage and decrease node generations when compared to their respective baselines.

Introduction

Classical planning is the problem of finding a sequence of actions leading from a specified initial state to some goal state. Whereas in classical planning outcomes of actions are uniquely determined, fully observable nondeterministic planning (FOND) permits actions whose outcomes are uncertain. Such nondeterministic actions can be used to model, e.g., the failure of an agent’s action. While this is often addressed by re-planning, strong cyclic plans—trial-and-error strategies—empower the agent to solve failure situations without re-planning.

Recently, research in classical planning has shifted towards techniques orthogonal to heuristics such as partial order reduction which has been transferred from computer aided verification (Valmari 1989; Godefroid 1995) to optimal deterministic planning (Alkhazraji et al. 2012). Further research aimed at improving the efficiency of stubborn set computation and determining a generalized definition of stubborn sets (Wehrle and Helmert 2014). We address the stubborn set method combined with two algorithms, Fast Incremental Planning (FIP) and LAO*.

Fast Incremental Planning is an algorithm for strong cyclic planning which solves FOND problems within multiple runs of an underlying classical planner (Kuter et al. 2008). Planner for Relevant Policies (PRP) combines this idea with a regression search to generalize the policy and substantially outperforms FIP (Muise, McIlraith, and Beck 2012). Our first step towards estimating the potential of stubborn sets for FOND planning is to use FIP with an underlying classical planner in combination with stubborn sets from classical planning. However, the main drawback of such determinization approaches is that they may find poor solutions, e.g., strong cyclic plans with high expected costs.

LAO* (Hansen and Zilberstein 2001), originally proposed to solve MDPs, is an algorithm for strong cyclic planning, which finds strong cyclic solutions in the nondeterministic state space. Using an admissible heuristic estimator, it finds strong cyclic plans of minimal expected costs. It has been shown that combining LAO* with pattern database heuristics (Mattmüller et al. 2010) is a successful approach to solving FOND problems. Our contribution is a stubborn set formalism for nondeterministic state spaces, that preserves strong cyclic plans. We evaluated both approaches, FIP with stubborn sets from classical planning and LAO* with our new formalism. Our results show that both approaches increase coverage and reduce node generations when compared to their respective baselines without stubborn sets.

Preliminaries

We use an $SAS^+$ based notation (Bäckström and Nebel 1993) to model fully observable nondeterministic planning problems. States of the world are described by a finite set of state variables $V$. Every variable $v \in V$ has an associated finite domain $D_v$ and an extended domain $D_v^+ = D_v \cup \{\bot\}$ where $\bot$ defines the undefined value. A partial state is a function $s$ with $s(v) \in D_v^+$ for all $v \in V$. We write $\text{vars}(s)$ for the set of all $v$ with $s(v) \neq \bot$. A partial state is a state if $\text{vars}(s) = V$.

Definition 1 (nondeterministic planning task). A nondeterministic planning task is a 4-tuple $\Pi = (V, O, s_0, s_*)$, where $V$ is a finite set of finite-domain variables, $O$ is a finite set of nondeterministic operators, $s_0$ is a state called the initial state and $s_*$ is a partial state called the goal. Each nondeterministic operator $o = (\text{Pre} \mid \text{Eff})$ has a partial state $\text{Pre}$ called precondition, a finite set of partial states $\text{Eff}$ and an associated non-negative number $\text{cost}(o)$ called its cost.

An operator $o$ is applicable in a state $s$ if $\text{Pre}$ is sat-
is to a goal state or to a state where the policy is defined. Policy is deterministic if |\(\mathcal{E}\)| does not revisit already visited states.

Definition 2 (weak plan, strong cyclic plan, strong plan). The determination of non-deterministic operator \(o\) is \(o^{[1]}, \ldots, o^{[k]}\) with every outcome \(o^{[i]} = \langle \text{Pre} \mid \{\text{eff}_1, \ldots, \text{eff}_k\}\rangle\). The all-outcomes determination of planning task \(\Pi = (\mathcal{V}, \mathcal{O}, s_0, s_*)\) is \(\Pi_{\text{det}} = (\mathcal{V}, \mathcal{O}_{\text{det}}, s_0, s_*)\) where \(\mathcal{O}_{\text{det}}\) is the set of all operator outcomes of \(\mathcal{O}\).

An operator is deterministic if |\(\mathcal{E}\)| ≥ 2. We say a planning task \(\Pi = (\mathcal{V}, \mathcal{O}, s_0, s_*)\) is deterministic if all of its operators are deterministic. We refer to all variables in the precondition of an operator \(o\) as \(\text{prev}(o)\) and to all variables in its effects as \(\text{eff}(o)\).

A solution to a FOND planning task \(\Pi\) with set of states \(S\) is a policy \(\pi: S \rightarrow \mathcal{O} \cup \{\bot\}\), which maps a state to an appropriate action or is undefined, e.g. \(\pi(s) = \bot\). Policy \(\pi\) is weak if it defines at least one path from the initial state to a goal state following it. It is closed if following it either leads to a goal state or to a state where the policy is defined. Policy \(\pi\) is proper if from every state visited following it there exists a path to a goal state following it. Policy \(\pi\) is acyclic if it does not revisit already visited states.

Definition 3 (interference of deterministic operators). Let \(\Pi = (\mathcal{V}, \mathcal{O}, s_0, s_*)\) be a planning task.

- A policy for \(\Pi\) is called a weak plan for \(\Pi\) if it is weak.
- A policy for \(\Pi\) is called a strong cyclic plan for \(\Pi\) if it is closed and proper.
- A policy for \(\Pi\) is called a strong plan for \(\Pi\) if it is closed proper and acyclic.

A weak plan is a sequence of actions which leads to the goal if all nondeterministic operator outcomes were deterministic. It corresponds to a plan in classical planning. A strong plan guarantees that after a maximum number of steps a goal state is reached. Strong cyclic planning relaxes that property requiring that the goal is reached within a finite sequence of actions. We want to emphasize that the nondeterminism in FOND planning is not necessarily the same as in model checking with nondeterministic models since unlike strong cyclic plans, counterexamples in model checking are linear sequences.

### Deterministic Stubborn Sets

The first step towards stubborn sets is the definition of operator interference. We follow the definition of Wehrle and Helmert (2014).

Definition 4 (deterministic strong stubborn set). Let \(\Pi = (\mathcal{V}, \mathcal{O}, s_0, s_*)\) be a deterministic planning task and \(s\) a state. A set \(\mathcal{T}_s \subseteq \mathcal{O}\) is a deterministic strong stubborn set (DSSS) in \(s\) if the following conditions hold:

1. \(\mathcal{T}_s\) contains a disjunctive action landmark in \(s\).
2. For all operators \(o \in \mathcal{T}_s\) with \(o \notin \text{app}(s)\), \(\mathcal{T}_s\) contains a necessary enabling set for \(o\) in \(s\).
3. For all operators \(o \in \mathcal{T}_s\) with \(o \in \text{app}(s)\), \(\mathcal{T}_s\) contains all operators that interfere with \(o\) in \(s\).

We use FIP combined with an underlying classical planner using deterministic stubborn sets. Solving FOND problems with classical planners can lead to costly strong cyclic plans. Although optimality does not play the major role in FOND planning, the possibility of finding arbitrarily bad solutions is undesirable. We show that exactly this might happen.

#### Example 1.

Consider a nondeterministic planning task \(\Pi = (\mathcal{V}, \mathcal{O}, s_0, s_*)\) with variables \(V = \{v_1, v_2\}\) and the following
operators:
• $o_1 = \langle v_1 = 0 \mid \{ v_1 \mapsto 1 \}, \{ v_2 \mapsto 2 \} \rangle$
• $o_2 = \langle v_1 = 1, v_2 = 0 \mid \{ \top \}, \{ v_1 \mapsto 0, v_2 \mapsto 2 \} \rangle$
• $o_3 = \langle v_1 = 0 \mid \{ v_2 \mapsto 2 \}, \{ v_2 \mapsto 1 \} \rangle$
• $o_4 = \langle v_1 = 0, v_2 = 1 \mid \{ \top \}, \{ v_2 \mapsto 2 \} \rangle$

As cost function we have $cost : \{ o_1 \mapsto 1, o_2 \mapsto 1000, o_3 \mapsto 2, o_4 \mapsto 1 \}$, the initial state is $s_0 = \{ v_1 \mapsto 0, v_2 \mapsto 0 \}$ and the goal $s_\ast = \{ v_2 \mapsto 2 \}$. Assume we perform a run of the FIP algorithm and its first weak plan would be $o_1^{[2]}$ inducing the fail-state $o_3^{[3]}(s_0) = 10$. In a subsequent weak plan search, the algorithm considers both outcomes of $o_2$ and adds them to the policy. This yields a clearly non-optimal strong cyclic plan, whereas the optimal solution consists of $o_3$ and $o_4$ (Figure 1). Applying PRP to this example gives the same solution, since regressing $o_1^{[2]}$ is ineffective.

### Non-deterministic Stubborn Sets

Reducing FOND problems to multiple classical planning problems sometimes leads to poor strong cyclic solutions since the individual runs of classical planners only guarantee good weak plans which are not always part of a good strong cyclic plans. To overcome this, it can be beneficial to plan in the nondeterministic state space e.g. with LAO* search (Hansen and Zilberstein 2001) which finds strong cyclic plans with minimum expected costs under certain assumptions. Planning in the nondeterministic state space needs new definitions of stubborn sets and operator interference since the former do not consider non-deterministic operators.

For a given nondeterministic planning problem $\Pi$, a straightforward approach would be to directly apply the original definition of strong stubborn sets on the all-outcome-determination of $\Pi$, and additionally, to add for every outcome $o^{[i]}$ of a non-deterministic operator $o$ every other outcome of $o$ in order to respect $o$'s non-deterministic nature. However, as the following example shows, such an approach is incomplete.

**Example 2.** Consider the following all-outcomes determination $\Pi_{det} = (\mathcal{V}, \mathcal{O}_{det}, s_0, s_\ast)$ of nondeterministic planning task $\Pi$ with variables $\mathcal{V} = \{v_1, v_2\}$ and the following operators:

- $o_1^{[1]} = \langle v_1 = 0 \mid \{ v_1 \mapsto 1 \}, v_2 = 0 \rangle$
- $o_1^{[2]} = \langle v_1 = 1 \mid \{ v_2 \mapsto 1 \}, v_2 = 2 \rangle$
- $o_2^{[1]} = \langle v_2 = 0 \mid \{ v_2 \mapsto 3 \}, v_2 = 2 \rangle$
- $o_3^{[2]} = \langle v_2 = 0 \mid \{ v_2 \mapsto 4 \}, v_2 = 2 \rangle$
- $o_{11} = \langle v_1 = 1, v_2 = 1 \mid \{ v_2 \mapsto 5 \} \rangle$
- $o_{12} = \langle v_1 = 1, v_2 = 2 \mid \{ v_2 \mapsto 5 \} \rangle$
- $o_{23} = \langle v_1 = 2, v_2 = 3 \mid \{ v_2 \mapsto 5 \} \rangle$
- $o_{24} = \langle v_1 = 2, v_2 = 4 \mid \{ v_2 \mapsto 5 \} \rangle$

The initial state is $s_0 = \{ v_1 \mapsto 0, v_2 \mapsto 0 \}$, and the goal is $s_\ast = \{ v_2 \mapsto 5 \}$. The set $\{o_{11}, o_{12}, o_{23}, o_{24}\}$ is a disjunctive action landmark in $s_0$ which we add to the candidate set $T_{s_0}$. As all operators in this set are inapplicable in $s_0$, we have to add a necessary enabling set for all of them. A valid choice for these necessary enabling sets is based on selecting the unsatisfied conditions $v_2 = 1, v_2 = 2, v_2 = 3$ and $v_2 = 4$ in the preconditions of $o_{11}, o_{12}, o_{23}, o_{24}$, respectively, and to add the determined operators that set these conditions to true. These achieving operators correspond to all outcomes of $o_2$ and $o_3$, which are applicable in $s_0$ but non-interfering with any operator not in $T_{s_0}$. Hence, we finally get $T_{s_0} = \{ o_{11}, o_{12}, o_{23}, o_{24} \} = \{ o_2 \} \cup \{ o_3 \} \cup \{ o_3 \} = \{ o_3 \}$.

However, $T_{s_0}$ is insufficient for our purpose because every strong plan from $s_0$ has to start with $o_1$. Depending on the nondeterministic outcome of $o_1 (v_1 = 1 \lor v_1 = 2)$, $o_2$ or $o_3$ can be applied to satisfy the precondition of an operator to reach the goal. In contrast, starting with $o_2$ and applying $o_1$ afterwards might lead to outcomes where no goal is reachable any more (e.g., $v_1 = 2$ and $v_2 = 2$). The analogous situation occurs when starting with $o_3$ and applying $o_1$ afterwards (Figure 2).

The core problem of our straightforward instantiation is that deterministic operator interference is an insufficient criterion for nondeterministic operators. Because changing the order of two non-interfering nondeterministic operators $o$ and $o'$ in a strong cyclic plan results in, e.g., outcomes of $o'$ getting prefixes of weak plans started by $o$. While this is not an issue for all weak plans which contain outcomes of both operators, it is problematic to those weak plans which start with an outcome of $o$ but do not contain an outcome of $o'$. A solution to this is to demand that such prefixes preserve the original weak plan which we address with the following property.

**Definition 5** (prefix-compatibility). Let $\Pi$ be a planning task and $\Pi_{det} = (\mathcal{V}, \mathcal{O}_{det}, s_0, s_\ast)$ its all-outcomes determination. Two operators $o_1, o_2 \in \mathcal{O}_{det}$ are prefix compatible if for all operator sequences $\pi_1$ and $\pi_2$:

- $o_1 \pi_1$ is a weak plan implies $o_2 o_1 \pi_1$ is also a weak plan and
- $o_2 \pi_2$ is a weak plan implies $o_1 o_2 \pi_2$ is also a weak plan

Intuitively, two operators $o_1$ and $o_2$ are prefix compatible if every weak plan starting with $o_1$ is preserved if we put $o_2$ to its front and vice versa. Equipped with prefix compatibility, we can formulate the definition of a stubborn set for the nondeterministic state space which has two additional rules compared to the DSS definition.

**Definition 6** (nondeterministic strong stubborn set). Let $\Pi = (\mathcal{V}, \mathcal{O}, s_0, s_\ast)$ be a nondeterministic planning task,

\[
\begin{align*}
\mathcal{V} & = \{ v_1, v_2 \}, \\
\mathcal{O} & = \{ o_1, o_2, o_3, o_4 \}, \\
\mathcal{S}_0 & = \{ v_1 \mapsto 0, v_2 \mapsto 0 \}, \\
\mathcal{S}_\ast & = \{ v_2 \mapsto 5 \}
\end{align*}
\]
\( \Pi_{det} = (\mathcal{V}, \mathcal{O}_{det}, s_0, s_s) \) its all-outcomes determination and \( s \) a state. A set \( T_s \subseteq \mathcal{O}_{det} \) is a nondeterministic strong stubborn set (NSSS) in \( s \) if the following conditions hold:

1. \( T_s \) contains a disjunctive action landmark in \( s \) for \( \Pi_{det} \).
2. For all operators \( o \in T_s \) with \( o \notin \text{app}(s) \), \( T_s \) contains a necessary enabling set for \( o \) in \( s \) for \( \Pi_{det} \).
3. For all operators \( o \in T_s \) with \( o \in \text{app}(s) \), \( T_s \) contains all operators that interfere with \( o \) in \( s \) for \( \Pi_{det} \).
4. For every outcome \( o[i] \in T_s \) of nondeterministic operator \( o \), \( T_s \) contains all operators that are not prefix compatible with \( o \).
5. For every outcome \( o[i] \in T_s \) of nondeterministic operator \( o \), \( T_s \) contains all other outcomes of \( o \).

**Proposition 1.** Nondeterministic strong stubborn sets preserve completeness for strong cyclic planning.

**Proof.** At first we show completeness for strong planning then we show it for strong cyclic planning. Let \( \pi \) be a strong plan from state \( s \) that induces weak plans \( \pi_i \) and \( \pi_j \), such that there is state \( s \) with \( \pi_i(s) = o[i] \) and \( \pi_j(s) = o[j] \). Weak plans \( \pi_i \) and \( \pi_j \) have the following structure: \( \pi_i = \alpha o[i] \beta_i \) and \( \pi_j = \alpha o[j] \beta_j \), where \( \alpha \) is a common operator sequence without outcomes of nondeterministic operators, and \( \beta_i, \beta_j \) contain also outcomes of nondeterministic operators respectively. State \( s \) is the branching point of \( \pi_i \) and \( \pi_j \). Let \( k_i \) be the smallest index such that operator \( o_k \in T_s \) is contained in the nondeterministic stubborn set \( T_s \), similarly for \( k_j \). We distinguish the following three cases:

1. \( o_k \in \alpha = o_1 \cdots o_n \). Clearly \( o_k = o_k \), \( o_k \) is applicable since otherwise a necessary enabling set has to be contained in \( T_s \) and at least one operator has to be applied before \( o_k \), contradicting the choice of \( k_i \). Since \( k_i \) is the smallest index such that \( o_k \in T_s \), \( o_k \) does not interfere with any operator of smaller index because otherwise an operator applied before \( o_k \) must be contained in \( T_s \). Also, this contradicts the choice of \( k_i \). Thus we can replace \( o_k \) by \( o_k, o_k \cdots o_k-1 o_k \cdot 1 \cdots o_n \).

2. \( o_k = o[i] \), \( o_k \notin \alpha \) since otherwise \( o_k \in \alpha \). Also, \( o_k \) cannot be in \( \beta_j \) because by the definition of the NSSS \( o[j] \in T_s \). It follows that \( o_k = o[j] \). Like in case (1) \( o_k \) is applicable and does not interfere with operators of smaller index for \( \pi_i \), the same holds for \( o_k \) and \( \pi_j \). Thus we can move the nondeterministic operator \( o \) to the front resulting in \( o[i] \beta_i o[j] \).

3. \( o_k \in \beta_i = o_{n+2} \cdots o_{n+m} \). \( o_k \) cannot be in \( \alpha \) since otherwise \( o_k \in \alpha \). Also, \( o_k \notin o[j] \) since otherwise by definition of the NSSS, \( o[i] \) would be included in \( T_s \), contradicting the choice of \( o_k \). Therefore \( o_k \in \beta_j \). Let \( s_1 \cdots s_n = \cdots s_{n+1} \cdots s_{n+m} \) be the states visited by \( \pi_i \). We know as in case (1) \( o_k \) does not interfere with operators of smaller index. Inductively it follows that \( o_k \) is applicable in \( s \). Also we know that \( o_k \) and \( o \) are prefix compatible since otherwise \( o \in T_s \). This means that \( o[i] o_{n+2} \cdots o_{n+m} \) is a weak plan from \( o_k(s) = o_k(o_n \cdots (o_1(s_0))) \). From the non-interference of \( o_k \) with operators of smaller index we get \( o_k(o_n \cdots (o_1(s_0))) = o_n(o_{n-1} \cdots (o_1(o_k(s_0)))) \).

Hence we can move \( o_k \) to the front of \( \pi_i \) and \( \pi_j \). If \( o_k \) is an outcome of a nondeterministic operator \( o' \) then it has a sibling \( o' \) which is the smallest index of weak plan \( \pi_i \). This case is covered by case (2) with \( \pi_i \) and \( \pi_j \).

Since a strong plan is a strong cyclic plan without cycles we just have to consider the effect of cycles. NSSS is state dependent but not path dependent, therefore revisiting some state \( s \) does not affect \( T_s \), concluding the proof.

Nondeterministic stubborn sets are in general not optimality preserving for strong cyclic planning since prefix compatibility leads to operators being added in front of other ones which can lead to solutions with higher expected costs.

**Approximating Prefix Compatibility**

The exact notion of prefix compatibility is intractable to compute because we would have to consider all weak plans. Therefore we outline how to find a sufficient criterion for prefix-compatibility. We define \( Dis(o) \) as the set of operator-variable pairs \( (o', v) \in \mathcal{O}_{det} \times \mathcal{V} \) such that \( o \) disables \( o' \) on variable \( v \) in any state. Further we define \( Neg(o) \) as the set of goal variables with which \( o \) conflicts, i.e., \( eff(o)[v] \neq s_i[v] \) for goal-related variables \( v \) on which \( o \) has an effect. If \( Dis(o_1) = Dis(o_2) \) and \( Neg(o_1) = Neg(o_2) \) then \( o_1 \) and \( o_2 \) are prefix compatible. The idea behind this is: if two operators \( o_1 \) and \( o_2 \) disable the same set of operators on the same variables, then every deterministic operator sequence starting with \( o_1 \) remains applicable if we append \( o_2 \) to its front. Also weak plans are preserved since \( o_1 \) and \( o_2 \) do not violate different goal variables.

In some cases, we can weaken this syntactic notion of prefix compatibility. Consider two non-interfering operators \( o_1 = \{Pre \ | \ \{eff_1\}\} \) and \( o_2 = \{Pre \ | \ \{eff_1, \cdots, eff_n\} \) if \( \sigma = \{s_1 \mapsto o_1, o_1(s) \mapsto o_2\} \) is a subsequence of a strong cyclic plan from state \( s \) then exchanging the order of \( o_1 \) and \( o_2 \) gives an equivalent subsequence since they induce the same set of states, i.e., \( o_1(o_2[s])(s) = o_2(o_1(s)) \) for all \( i \leq n \). Therefore if two such operators do not interfere, it suffices to check \( Dis(o_1) \subseteq Dis(o_2) \) and \( Neg(o_1) \subseteq Neg(o_2) \).

Sometimes nondeterministic operators contain only one nontrivial effect, i.e., an operator \( o = \{Pre \ | \ \{eff_1\}, \{\top\} \) for every weak plan \( o[i] \pi \) from state \( s \), it exists a finite sequence \( \sigma = o[i] \cdots o[2], \) repeated applications of \( o \)’s trivial effect, such that \( o \sigma \pi \) is also a weak plan from \( s \). Thus, every operator being a prefix of \( o[i] \pi \) preserving the weak plan, does also preserve \( o \sigma \pi \). Such operators are therefore trivially prefix compatible to any other deterministic operator.

**Efficient Computation**

As nondeterministic stubborn sets leave open how the disjunctive action landmark and the necessary enabling sets were chosen, the pruning power of stubborn sets depends highly on these design choices. We outlined how prefix compatibility can be syntactically addressed. However, for
applicable nondeterministic operators with more than one nontrivial effect, in the stubborn set we have to add both the interfering and the non-prefix compatible operators. This leads to many operators being added to the stubborn set. It is therefore reasonable to avoid applicable nondeterministic operators with more than one nontrivial effect from being added to the stubborn set. Let nontrivial be the set of operators with more than one nontrivial effect. Our intention is to exclude applicable operators of nontrivial from being added to the stubborn set. We address this by computing a weight whenever we have to add a DAL or NES to the stubborn set. We calculate a weight for each DAL or NES and chose the DAL or NES with lowest weight according to:

\[
weight(o, s, T_s) = \begin{cases} 
\infty, & \text{if } o \in \text{app}(s) \land o \in \text{nontrivial} \\
K, & \text{if } o \in \text{app}(s) \land o \notin \text{nontrivial} \\
1, & \text{otherwise}
\end{cases}
\]

where \(o\) is an operator not in stubborn set \(T_s\) and \(K\) a nonzero natural number. Our exclude strategy is an extension of a strategy presented by Laarman et al. (2013) which penalizes applicable operators not in the current candidate stubborn set. A coarser strategy towards prefix compatibility for nontrivial operators is to simply assume that an applicable nontrivial operator is not prefix compatible to all other operators. On par with the exclude strategy this is feasible since it avoids the costly computation of the disabling relation.

A Tighter Envelope

Active operators (Chen and Yao 2009; Wehrle et al. 2013) approximate the set of operators which can be part of any weak plan from some state using domain transitions graphs (DTGs). From a more general point of view, Wehrle et al. (2013) denote subsets which preserve at least one weak plan (DTGs). Combining a tight envelope with stubborn sets may not only exclude operators which are not in envelope \(E\) from the stubborn set but also prevent cascades from being added to the stubborn set. Of course, the active operators can also be used for strong cyclic planning since strong cyclic plans consist of multiple weak plans. We additionally exploit the structure of strong cyclic plans and obtain a tighter envelope.

A part-of-a-plan operator \(o \in \mathcal{O}\) in \(s\) is a deterministic operator that is contained in some weak plan starting from \(s\). This notion is intractable to compute so we have to find a sufficient criterion for it.

Definition 7 (active operator). Let \(\Pi = \langle \mathcal{V}, \mathcal{O}, s_0, s_\ast \rangle\) be a deterministic planning task. An active operator \(o \in \mathcal{O}\) in a state \(s\) is an operator that satisfies the following conditions:

1. For every variable \(v \in \text{prevars}(o)\), there is a path in \(DTG(v)\) from \(s[v] \to \text{pre}(o)[v]\), and also from \(\text{pre}(o)[v] \to \text{goal value} s[v]\) if \(v\) is goal-related.
2. For all \(v \in \text{effvars}(o) \cap \text{vars}(s_\ast)\) there is a path in \(DTG(v)\) from \(\text{eff}(o)[v] \to s[s_v]\).

Intuitively, the definition states that an operator is active if it is part of some weak plan from the corresponding abstracted state in every singleton abstraction of \(\Pi\). Thus, a part-of-a-plan operator is always active but not vice versa.

A nondeterministic part-of-a-plan operator \(o \in \mathcal{O}\) in \(s\) is an operator that is contained in some strong plan \(\pi\) starting from \(s\). Like for the part-of-a-plan operators this is intractable to compute.

Proposition 2. Let \(o\) be an applicable nondeterministic operator and \(o[\pi]\) one of its outcomes. If \(o[\pi]\) is inactive in state \(s\) then \(o\) cannot be part of any strong cyclic plan from \(s\).

Proof. We show this by contradiction. If there were a strong cyclic plan \(\pi\) from \(s\), such that \(\pi(s) = o\) for some state \(s\) but \(o[\pi]\) is no part-of-a-plan operator in \(s\). Let further \(\sigma = o_1 \cdots o_n\) be a deterministic operator sequence applicable in \(s\) that leads to \(s\). Since \(o[\pi]\) is no part-of-a-plan operator in \(s\), no weak plan from \(s\) does contain \(o[\pi]\). Therefore \(\sigma o[\pi] \pi\) for an arbitrary operator sequence \(\pi\) cannot be a weak plan from \(s\). This implies that \(\pi\) is no weak plan from \(o[\pi]\). Thus \(\pi\) is not proper since \(o[\pi]\) is reachable following \(\pi\) but there is no goal state reachable from \(o[\pi]\). This contradicts \(\pi\) being a strong plan. It follows that if any outcome of a nondeterministic operator \(o\) is not active in \(s\), then \(o\) cannot be part of a strong cyclic plan from \(s\).

We denote our new envelope by nondeterministic active envelope. It can be used for both the DSSS and NSSS.

Experimental Evaluation

We focused our experimental evaluation on the following two configurations:

1. FIP combined with DSSSs
2. LAO* combined with NSSSs

We further investigated the impact of different envelopes: full, active, nondeterministic active (Table 1 and Table 2). Also, we varied the approximation of prefix compatibility for the NSSS approach (Table 3). We differentiate between the approach which assumes every nontrivial operator is not prefix compatible with all other operators (no prefix) and the approach where prefix compatibility is syntactically approximated (syntactic). For the DSSS, disjunctive action landmarks and necessary enabling sets were computed using the laarman strategy. For the NSSS we used the exclude strategy. The interference relation for both the DSSSs and NSSSs is entirely precomputed which is also true for the achievers, the NSSSs need the additional precomputation of the disabling relation. For the underlying classical planner of FIP, we used greedy best first search. As heuristic estimator, we chose the FF heuristic (Hoffmann and Nebel 2001) for all approaches.

We evaluated both stubborn set approaches on all FOND domains of the IPC-2008 and variations of these. Furthermore we added two domains from probabilistic planning to our benchmark set. All experiments were conducted on a server equipped with AMD Opteron 2.3 GHz CPUs. We set

\footnote{First-Responders-new consists of larger instances of the First-Responders domain. Forest-new is taken by Muise, McIlraith, and Beck (2012). Tidyup is the Mobile Manipulation domain of Hertle et al. (2014) adapted for FOND planning. Earth-Observation was introduced by Aldinger and Lühr (2013).}
Our experiments yield the following insights:

- **NSSS approach**: LAO* search combined with NSSS does also clearly outperform its baseline in terms of coverage and node expansions. But in contrast to the DSSS approach, node generations are not as drastically reduced. Since the LAO* algorithm produces noticeably more overhead per node than the FIP algorithm, reducing a single node has greater impact for the LAO* algorithm than for FIP. The coverage increase is most evident in the domain Forest-new where three additional instances are solved and First-Responders-new with two additionally solved instances. The loss of two instances in First-Responders-new is caused by heuristic tie-breakings (100.7% node expansions). A blind search without stubborn set does not solve any problem in this domain, whereas in combination with NSSS, it solves the smallest instance. Furthermore, active envelope and nondeterministic active envelope are beneficial in terms of pruning power. We again see the empirical dominance of nondeterministic active envelope over active envelope supporting the theoretical results.

- **Prefix Compatibility**: In terms of pruning power, the approach without prefix compatibility for the nontrivial operators is dominated by the syntactic approximation of prefix compatibility. Except in the domain First-Responders-new more nodes are generated because of a single instance which is caused by heuristic tie-breaking (100.45% node expansions). The increased pruning power is reflected in better coverage for First-Responders-new and Forest-new where two hard additional instances are solved respectively.

### Results

Our experiments yield the following insights:

- **DSSS approach**: FIP combined with DSSS significantly increases coverage and reduces node expansions. This is particularly pronounced in the domains Forest and Forest-new. DSSS with active envelope solves five additional problem instances of the Tidyup domain which has—in contrast to all other domains—applicable inactive operators permitting immense pruning (0.1% nodes generated). In all other domains the effect of active envelope and also of nondeterministic active envelope is negligible. In First-Responders and First-Responders-new the DSSS approach loses one instance because of heuristic tie-breaking. Also, it loses two instances in Earth-Observation and one in Blocksworld. The instances in Earth-Observation result from the vast number of generated states. In Blocksworld the DSSS approach fails to solve the hardest problem solved by the baseline which solves it close to the time limit (1564s out of 1800s).

### Acknowledgments

This work was partly supported by the DFG as part of the SFB/TR 14 AVACS and by the Swiss National Science Foundation (SNSF) as part of the project “Automated Reformulation and Pruning in Factored State Spaces (ARAP)”.

### Table 1: Comparison of plain FIP with FIP using DSSS and different envelopes: full, active (DACT), nondeterministic active (NACT). Nodes of DACT, NACT in % of plain FIP.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Coverage</th>
<th>Node Generations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FIP</td>
<td>DSSS</td>
</tr>
<tr>
<td>FR(5)</td>
<td>74</td>
<td>+1</td>
</tr>
<tr>
<td>FR-New(91)</td>
<td>75</td>
<td>-1</td>
</tr>
<tr>
<td>Forest-New(90)</td>
<td>16</td>
<td>+4</td>
</tr>
<tr>
<td>Forest(90)</td>
<td>13</td>
<td>+4</td>
</tr>
<tr>
<td>Earth(40)</td>
<td>35</td>
<td>-2</td>
</tr>
<tr>
<td>Tidyup(10)</td>
<td>5</td>
<td>±0</td>
</tr>
<tr>
<td>Tidyupworld(40)</td>
<td>3</td>
<td>±0</td>
</tr>
<tr>
<td>BW(30)</td>
<td>25</td>
<td>-1</td>
</tr>
<tr>
<td>Faults(55)</td>
<td>55</td>
<td>±0</td>
</tr>
<tr>
<td>Overall</td>
<td>301</td>
<td>+3</td>
</tr>
</tbody>
</table>

### Table 2: Comparison of plain LAO* with LAO* using NSSS and different envelopes: full, active (DACT), nondeterministic active (NACT). Nodes of DACT, NACT in % of plain LAO*.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Coverage</th>
<th>Node Generations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LAO*</td>
<td>NSSS</td>
</tr>
<tr>
<td>FR(5)</td>
<td>59</td>
<td>+1</td>
</tr>
<tr>
<td>FR-New(91)</td>
<td>19</td>
<td>-2</td>
</tr>
<tr>
<td>Forest-New(90)</td>
<td>3</td>
<td>+3</td>
</tr>
<tr>
<td>Forest(90)</td>
<td>6</td>
<td>+1</td>
</tr>
<tr>
<td>Earth(40)</td>
<td>30</td>
<td>±0</td>
</tr>
<tr>
<td>Tidyup(10)</td>
<td>9</td>
<td>±0</td>
</tr>
<tr>
<td>Tidyupworld(40)</td>
<td>6</td>
<td>±0</td>
</tr>
<tr>
<td>BW(30)</td>
<td>21</td>
<td>±0</td>
</tr>
<tr>
<td>Faults(55)</td>
<td>54</td>
<td>±0</td>
</tr>
<tr>
<td>Overall</td>
<td>206</td>
<td>+3</td>
</tr>
</tbody>
</table>

### Table 3: Comparison of no prefix compatibility for nontrivial operators with syntactic approximation. We grouped the domains where coverage and node generations are equal. Nodes of syntactic in % of no prefix approach.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Coverage</th>
<th>Node Generations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>no prefix</td>
<td>syntactic</td>
</tr>
<tr>
<td>FR(5)</td>
<td>59</td>
<td>+2</td>
</tr>
<tr>
<td>FR-New(91)</td>
<td>17</td>
<td>+1</td>
</tr>
<tr>
<td>Forest-New(90)</td>
<td>6</td>
<td>+1</td>
</tr>
<tr>
<td>Forest(90)</td>
<td>7</td>
<td>-1</td>
</tr>
</tbody>
</table>
References


