RC25676 (WAT1805-052) May 14, 2018
Computer Science

IBM Research Report

Red-Black Heuristic for Planning Tasks with Conditional Effects

Michael Katz
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598 USA

Red-Black Heuristic for Planning Tasks with Conditional Effects

Michael Katz
IBM Research
Yorktown Heights, NY, USA
michael.katz1@ibm.com

Abstract

Red-black planning is the state-of-the-art approach to satisficing classical planning. A planner *Mercury*, empowered by the red-black planning heuristic, was the runner-up of the International Planning Competition (IPC) 2014. The exceptional performance of *Mercury* is amplified by the fact that conditional effects, a feature required to be supported in the competition, were handled by the planner in a trivial way, by simply compiling them away. Conditional effects, however, are important for classical planning, and many domains require them for efficient modeling.

Consequently, we herein investigate the possibility of handling conditional effects directly in the red-black planning heuristic function, extending the algorithm for computing red-black plans to the conditional effects setting. We show empirically that red-black planning heuristics that handle conditional effects natively favorably compete with the variant that compiles this feature away.

Introduction

Delete relaxation heuristics have played a key role in the success of satisficing planning systems (Bonet and Geffner 2001; Hoffmann and Nebel 2001; Richter and Westphal 2010). A well-known pitfall of delete relaxation is its inability to account for repetive achievements of facts. It has thus been an actively researched question from the outset how to take *some* deletes into account, e. g. (Fox and Long 2001; Gerevini, Saetti, and Serina 2003; Helmert 2004; Helmert and Geffner 2008; Baier and Botea 2009; Cai, Hoffmann, and Helmert 2009; Haslum 2012; Keyder, Hoffmann, and Haslum 2012). Red-black planning framework (Domshlak, Hoffmann, and Katz 2015), where a subset of *red* state variables takes on the relaxed value-accumulating semantics, while the other *black* variables retain the regular semantics, introduced a convenient way of interpolating between fully relaxed and regular planning.

Katz, Hoffmann, and Domshlak (2013b) introduced the red-black framework and conducted a theoretical investigation of tractability of both plan existence and plan generation. Among other, they found that *reversibility* (Chen and Giménez 2010) plays a major role in making red-black plan generation tractable. Following up on this, exploiting the notion of *invertibility*, Katz, Hoffmann, and Domshlak (2013a) devised practical *red-black plan heuristics*, non-admissible heuristics generated by repairing fully delete-relaxed plans into red-black plans. Observing that this technique often suffers from dramatic over-estimation incurred by following arbitrary decisions taken in delete-relaxed plans, Katz and Hoffmann (2013) refined the approach to rely less on such decisions, yielding a more flexible algorithm delivering better search guidance. Finally, Katz and Hoffmann (2014b) presented *red-black DAG heuristics* for a tractable fragment characterized by *DAG black causal graphs* and devised some enhancements targeted at making the resulting red-black plans executable in the real task, stopping the search if they succeed in reaching the goal. Red-black DAG heuristics are in the heart of the *Mercury* planner (Katz and Hoffmann 2014a), the runner-up of the sequential satisficing track in the International Planning Competition (IPC 2014). All red-black heuristics to this day, however, are defined for a SAS$^+$ fragment without conditional effects, despite of conditional effects being a main feature in the domains of IPC 2014. *Mercury* planner handles conditional effects by simply compiling them away in a straighforward fashion, multiplying-out the actions (Nebel 2000). Obviously, the number of actions grows exponentially, and thus the straight forward compiling away does not scale well. Nebel (2000) presents an alternative compilation, that does not lead to an exponential blow-up in the task size. This compilation, however does not preserve the delete relaxation. Thus, several delete relaxation based heuristics were adapted to natively support conditional effects (Haslum 2013; Röger, Pommerening, and Helmert 2014).

In this work we extend the red-black planning framework to tasks with conditional effects. To that end, we generalize the definition of *invertibility* to support conditional effects. We then show that the fragment of red-black planning characterized by *DAG black causal graphs* remains tractable in the presence of conditional effects. Further, we show how an existing algorithm for solving tasks belonging to this fragment can be adapted to handle conditional effects. Finally, we empirically show the added value of handling the conditional effects directly in the heuristic over compiling them away. We conclude the paper with a discussion of our results and a future work.

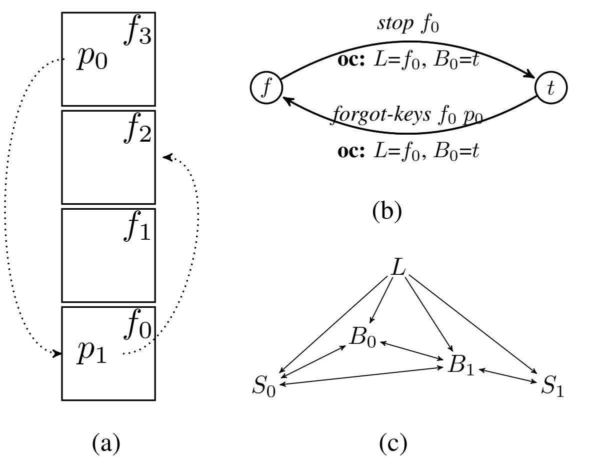
Background

In order to handle classical planning tasks with conditional effects, we consider the *red-black planning finite-domain representation (RB)* framework (Domshlak, Hoffmann, and Katz 2015), a generalization of both the finite-domain representation and the monotonic finite-domain representation formalisms. We extend the formalism of RB to handle actions with conditional effects. A **red-black (RB)** planning task is a tuple $\Pi = \langle \mathcal{V}^{\mathsf{B}}, \mathcal{V}^{\mathsf{R}}, O, s_0, s_\star \rangle$. $\mathcal{V}^{\mathsf{B}}$ is a set of *black state variables* and $\mathcal{V}^{\mathsf{R}}$ is a set of *red state variables*, where $\mathcal{V}^{\mathsf{B}} \cap \mathcal{V}^{\mathsf{R}} = \emptyset$ and each $v \in \mathcal{V} := \mathcal{V}^{\mathsf{B}} \cup \mathcal{V}^{\mathsf{R}}$ is associated with a finite domain $\mathcal{D}(v)$. The *initial state* s_0 is a complete assignment to $\mathcal{V}$, the *goal* $s_\star$ is a partial assignment to $\mathcal{V}$. Each action o is a pair $\langle pre(o), \mathit{effs}(o) \rangle$, where $pre(o)$ is a partial assignment to $\mathcal{V}$ called *precondition* and $\mathit{effs}(o)$ is a set of *effects*. Each effect $e \in \mathit{effs}(o)$ is a tuple $\langle cond, v, \vartheta \rangle$, where $cond$ is a partial assignment called *effect condition*, $v \in \mathcal{V}$ is the *effect variable*, and $\vartheta \in \mathcal{D}(v)$ is the *effect value*. We often refer to (partial) assignments as sets of *facts*, i. e., variable-value pairs $v = d$. For a partial assignment p, $vars(p)$ denotes the subset of $\mathcal{V}$ instantiated by p. For $\mathcal{V}' \subseteq vars(p)$, $p[\mathcal{V}']$ denotes the value of $\mathcal{V}'$ in p. For the sake of readability, by $vars(\mathit{effs}(o))$ we denote the subset of $\mathcal{V}$ that appear in $\mathit{effs}(o)$. Also, for the sake of readability, by $\mathit{effs}(o)[\mathcal{V}']$ we refer to the subset of conditional effects that affect variables in $\mathcal{V}'$.

A state s assigns each $v \in \mathcal{V}$ a non-empty subset $s[v] \subseteq \mathcal{D}(v)$, where $|s[v]| = 1$ for all $v \in \mathcal{V}^{\mathsf{B}}$. A state s *agrees* with the partial passignment p, denoted by $s \models p$, if $p[v] \in s[v]$ for all $v \in vars(p)$. An action o is applicable in state s if $s \models pre(o)$. An effect $\langle cond, v, \vartheta \rangle \in \mathit{effs}(o)$ *fires* in state s if $s \models cond$. Applying o in s changes the value of v for all firing effects $\langle cond, v, \vartheta \rangle \in \mathit{effs}(o)$ as follows. If $v \in \mathcal{V}^{\mathsf{B}}$, the value is changed to $\{\vartheta\}$. Otherwise (if $v \in \mathcal{V}^{\mathsf{R}}$), the new value of v is $s[v] \cup \{\vartheta\}$. By $s[\![\langle o_1, \dots, o_k \rangle]\!]$ we denote the state obtained from sequential application of $o_1, \dots, o_k$. An action sequence $\langle o_1, \dots, o_k \rangle$ is a *plan* if $s_\star[v] \in s_0[\![\langle o_1, \dots, o_k \rangle]\!][v]$ for all $v \in vars(s_\star)$.

Π is a **finite-domain representation (FDR)** planning task if $\mathcal{V}^{\mathsf{R}} = \emptyset$, and is a **monotonic finite-domain representation (MFDR)** planning task if $\mathcal{V}^{\mathsf{B}} = \emptyset$. Plans for MFDR tasks (i. e., for delete-relaxed tasks) can be generated in polynomial time. A key part of many satisficing planning systems is based on exploiting this property for deriving heuristic estimates, via delete-relaxing the task at hand. Generalizing this to red-black planning, the **red-black relaxation** of an FDR task Π relative to $\mathcal{V}^{\mathsf{R}}$ is the RB task $\Pi^{*+}_{\mathcal{V}^{\mathsf{R}}} = \langle \mathcal{V} \setminus \mathcal{V}^{\mathsf{R}}, \mathcal{V}^{\mathsf{R}}, O, s_0, s_\star \rangle$. A plan for $\Pi^{*+}_{\mathcal{V}^{\mathsf{R}}}$ is a **red-black plan** for Π, and the length of a shortest possible red-black plan is denoted $h^{*+}_{\mathcal{V}^{\mathsf{R}}}(\Pi)$. For arbitrary states s, $h^{*+}_{\mathcal{V}^{\mathsf{R}}}(s)$ is defined via the RB task $\langle \mathcal{V} \setminus \mathcal{V}^{\mathsf{R}}, \mathcal{V}^{\mathsf{R}}, O, s, s_\star \rangle$. If $\mathcal{V}^{\mathsf{R}} = \mathcal{V}$, then red-black plans are **relaxed plans**, and $h^{*+}_{\mathcal{V}^{\mathsf{R}}}$ coincides with the optimal delete relaxation heuristic h^+.

We use a slightly modified miconic-simpleadl task with two passengers as our running example. An elevator moves between four floors and need to move passengers from their original floors to their destination floors. Passenger p_0 is originally at floor f_3 and needs to get to floor f_0. Passenger p_1 starts from floor f_0 and goes to floor f_2. The example is depicted in Figure 1(a), with floors, passengers and their initial and goal locations shown. There is one variable for elevator location, L, with initial value f_0, and two variables for each passenger, encoding whether the passenger was boarded, B_i and whether she was served, S_i. There are actions changing the value of variable L, *up* f_i f_j and *down* f_i f_j. These actions have one unconditional effect each. In addition, there are actions that stop at a floor, picking up passengers from their origin floors and embarking passengers to their destinations. The *stop* f_i actions are as follows.



p_0
f_3
f_2
f_1
f_0
p_1
(a)
stop f_0
oc: L=f_0, B_0=t
f
t
forgot-keys f_0 p_0
oc: L=f_0, B_0=t
(b)
L
B_0
B_1
S_0
S_1
(c)

Figure 1: An example (a), a domain transition graph of a variable S_0 (b), and the causal graph (c).

$$\mathit{stop}\ f_0 = \left\langle \{L{=}f_0\}, \left\{ \begin{array}{r} \langle \emptyset, B_0, f \rangle, \\ \langle \{B_0{=}t\}, S_0, t \rangle, \\ \langle \{S_1{=}f\}, B_1, t \rangle \end{array} \right\} \right\rangle,$$

$$\mathit{stop}\ f_2 = \langle \{L{=}f_2\}, \{\langle \emptyset, B_1, f \rangle, \langle \{B_1{=}t\}, S_1, t \rangle\} \rangle, \text{ and}$$

$$\mathit{stop}\ f_3 = \langle \{L{=}f_3\}, \{\langle \{S_0{=}f\}, B_0, t \rangle\} \rangle.$$

Note that there are only actions *stop* f_0, *stop* f_2, and *stop* f_3. There is no action *stop* f_1, since the floor f_1 is neither origin nor destination of any passenger.

In addition, there are actions *forgot-keys* f p with one unconditional effect each, taking the embarked passenger back to the elevator on her destination floor. These actions are as follows.

$$forgot\text{-}keys\ f_0\ p_0 = \langle \{L{=}f_0, B_0{=}t, S_0{=}t\}, \{\langle \emptyset, S_0, f \rangle\} \rangle,$$

$$forgot\text{-}keys\ f_2\ p_1 = \langle \{L{=}f_2, B_1{=}t, S_1{=}t\}, \{\langle \emptyset, S_1, f \rangle\} \rangle.$$

Focusing at action *stop* f_0, a straight forward approach of handling conditional effects would multiply out the variables in conditions of the effects. As a result, we obtain four actions:

- $\langle \{L{=}0, B_0{=}f, S_1{=}f\}, \{\langle \emptyset, B_0, f \rangle, \langle \emptyset, B_1, t \rangle\} \rangle$,
- $\langle \{L{=}0, B_0{=}t, S_1{=}f\}, \{\langle \emptyset, B_0, f \rangle, \langle \emptyset, B_1, t \rangle, \langle \emptyset, S_0, t \rangle\} \rangle$,
- $\langle \{L{=}0, B_0{=}f, S_1{=}t\}, \{\langle \emptyset, B_0, f \rangle\} \rangle$, and
- $\langle \{L{=}0, B_0{=}t, S_1{=}t\}, \{\langle \emptyset, B_0, f \rangle, \langle \emptyset, S_0, t \rangle\} \rangle$.

A real optimal plan has length 6 (*up* f_0 f_3, *stop* f_3, *down* f_3 f_0, *stop* f_0, *up* f_0 f_2, *stop* f_2), a relaxed plan has length 5 (no need to go back down to f_0). If we paint L black, then $h^{*+}_{\mathcal{V}^{\mathsf{R}}}(s_0) = 6$ as desired.

Tractable fragments of red-black planning have been identified using standard structures, **domain transition graph** and **causal graph** (Helmert 2006). The **domain transition graph** $DTG_\Pi(v)$ of a variable $v \in \mathcal{V}$ is a labeled digraph with vertices $\mathcal{D}(v)$. The graph has an arc (d, d') induced by action o if $\langle cond, v, d' \rangle \in \mathit{effs}(o)$, and either (i) $pre(o)[v] = d$ or $cond[v] = d$, or (ii) $v \notin vars(pre(o)) \cup vars(cond)$. The arc is labeled with its induced action o and its **outside condition** $pre(o)[\mathcal{V} \setminus \{v\}] \cup cond[\mathcal{V} \setminus \{v\}]$. In contrast to the case of no conditional effects, here we cannot know in advance which effects will fire, and thus the notion of **outside effect** for an edge in the domain transition graph is not well defined. The domain transition graph of the variable S_0 in our example is shown in Figure 1(b). The action labels are above the arcs and the outside conditions (marked by **oc**) are below the arcs.

The **causal graph** CG_Π of Π is a digraph with vertices $\mathcal{V}$. An arc (v, v') is in CG_Π if $v \neq v'$ and there exists an action $o \in O$ such that either (i) the domain transition graph $DTG_\Pi(v')$ of v' has some arc labeled with outside condition on the variable v, or (ii) both $v, v' \in vars(\mathit{effs}(o))$.

The **black causal graph** CG^{B}_Π of Π is the sub-graph of CG_Π induced by $\mathcal{V}^{\mathsf{B}}$. Figure 1(c) depicts the causal graph of the example task. If we paint only L black, then the black causal graph is arc-empty. If we paint also S_0 and S_1 black, then the black causal graph is a DAG.

Invertibility

The notion of invertibility introduces a sufficient criteria for tractability of red-black planning for tasks with acyclic black causal graphs (Katz, Hoffmann, and Domshlak 2013a; Katz and Hoffmann 2013; Domshlak, Hoffmann, and Katz 2015). The definition for tasks without conditional effects is as follows. An arc (d, d') is **relaxed side effects invertible**, **RSE-invertible** for short, if there exists an arc (d', d) with outside condition $\phi' \subseteq \phi \cup \psi$ where ϕ and ψ are the outside condition respectively outside effect of (d, d'). A variable v is RSE-invertible if all arcs in $DTG_\Pi(v)$ are RSE-invertible, and an RB task is RSE-invertible if all its black variables are.

The motivation behind the defition is as follows. If the black causal graph is acyclic, every action affects at most one black variable. Since red variables accumulate their values, the only effect we need to invert is the black one. This corresponds to inverting a single arc (d, d') in a domain transition graph. Thus, we need to ensure that for every arc (d, d') traversed in the domain transition graph, there exist a corresponding inverse arc (d', d). That arc must be traversable after (d, d'). Furthermore, since the outside condition ϕ of (d, d') is not changed by traversing the arc, and the outside effect ψ consists of red variables only, and thus can only accumulate new values by traversing the arc, both ϕ and ψ will remain true and can be used as conditions for the inverse arc.

In the presence of conditional effects we need to cosider which effects fire, and thus the definition of **RSE-invertible** arcs is adapted as follows.

Definition 1 *Let $v \in \mathcal{V}$ be some variable and o be some action affecting v. Let (d, d') be some arc in the domain transition graph $DTG_\Pi(v)$ of v, induced by $\langle cond, v, d' \rangle \in \mathit{effs}(o)$ of action o and let ϕ be its outside condition. The arc (d, d') is **relaxed side effects invertible**, **RSE-invertible** for short, if there exists an induced by the action o' arc (d', d) with outside condition ϕ', such that $\phi' \subseteq \phi \cup \psi$, where $\psi = \{v' = d'' \mid \langle cond', v', d'' \rangle \in \mathit{effs}(o), v' \neq v, cond' \subseteq \phi\}$.*

In words, the outside condition of the reverting arc should either hold before traversing the reverted arc or be achieved as a side effect of traversing the arc. For latter, we can consider only the effects that surely fire.

As before, a variable v is RSE-invertible if all arcs in $DTG_\Pi(v)$ are RSE-invertible, and an RB task is RSE-invertible if all its black variables are. In what follows, we show that the main theorem of Katz, Hoffmann, and Domshlak (2013a), which enabled devising practical red-black heuristics holds also for tasks with conditional effects.

Theorem 1 *Any RSE-invertible RB task with acyclic black causal graph is reversible.*

We follow the proof of Katz, Hoffmann, and Domshlak (2013a) and show that every action application can be undone. Specifically, we show that given a state s and an action o applicable to s, from the state $s[\![\langle o \rangle]\!]$ we can reach a state s' so that $s'[\mathcal{V}^{\mathsf{B}}] = s[\mathcal{V}^{\mathsf{B}}]$. If all variables affected by o are red, $s' := s[\![\langle o \rangle]\!]$ fits the requirement. Otherwise, o affects exactly one black variable v. Let (d, d') be the arc in $DTG_\Pi(v)$ that corresponds to the effect $\langle cond, v, d' \rangle \in \mathit{effs}(o)$ fired in s, let (d', d) be the inverse arc, and let o' be the action that induces (d', d) with outside condition ϕ' as in Definition 1. Then o' is applicable in $s[\![\langle o \rangle]\!]$: Using the notations from above, $pre(o') \subseteq \phi' \cup \{(v, d')\}$, where $\phi' \subseteq \phi \cup \psi$. As discussed above, both ϕ and ψ are true in $s[\![\langle o \rangle]\!]$. Clearly, $s' := s[\![\langle o, o' \rangle]\!]$ has the required property. This concludes the proof of Theorem 1.

In the example, variable L is RSE-invertible, since *up* and *down* actions invert each other. The variables B_0 and B_1 are not RSE-invertible, since edges that correspond to *stop* actions can be inverted only by other *stop* actions. These pairs of *stop* actions correspond to boarding and embarking the passenger, and therefore the elevator location value would not match. The variables S_0 and S_1 are RSE-invertible, as can be seen in Figure 1(b). Note that while the edge from t to f coressponds to a non-conditional effect and thus the outside condition comes entirely from the precondition, the edge from f to t is induced by a conditional effect $\langle \{B_0{=}t\}, S_0, t \rangle$, and thus the outside condition comes partially from the precondition and partially from the effect condition.

Red-Black DAG Heuristics

Katz and Hoffmann (2013) provide an algorithm for RSE-invertible RB tasks with acyclic black causal graphs and no

```
Algorithm : REDBLACKPLANNINGCE($\Pi, R^+$)
  main
    // $\Pi = \langle \mathcal{V}^{\mathsf{B}}, \mathcal{V}^{\mathsf{R}}, O, s_0, s_\star \rangle$
    global $R$,  $B \leftarrow \emptyset$,  $\pi \leftarrow \langle\rangle$
    UPDATE()
    while $R \not\supseteq R^+$
      do
        $O' = \{\langle o, c\rangle \mid o \in O, \langle c, v, \vartheta\rangle \in \mathit{effs}(o),$
                      $\mathit{pre}(o) \cup c \subseteq B \cup R,$
                      $\langle v, \vartheta\rangle \in (R^+ \setminus R)\}$
        Select $\langle o, c\rangle \in O'$
        if $\mathit{pre}(o)[\mathcal{V}^{\mathsf{B}}] \cup c[\mathcal{V}^{\mathsf{B}}] \not\subseteq s_0[\![\pi]\!]$
          then $\pi \leftarrow \pi \circ$ ACHIEVE($\mathit{pre}(o)[\mathcal{V}^{\mathsf{B}}] \cup c[\mathcal{V}^{\mathsf{B}}]$)
        $\pi \leftarrow \pi \circ \langle o\rangle$
        UPDATE()
    if $s_\star[\mathcal{V}^{\mathsf{B}}] \not\subseteq s_0[\![\pi]\!]$
      then $\pi \leftarrow \pi \circ$ ACHIEVE($s_\star[\mathcal{V}^{\mathsf{B}}]$)
    return $\pi$

  procedure UPDATE()
    $R \leftarrow s_0[\![\pi]\!][\mathcal{V}^{\mathsf{R}}]$
    $B \leftarrow B \cup s_0[\![\pi]\!][\mathcal{V}^{\mathsf{B}}]$
    for $v \in \mathcal{V}^{\mathsf{B}}$, ordered topologically by the black causal graph
      do $B \leftarrow B \cup \mathit{DTG}_\Pi(v)|_{R \cup B}$

  procedure ACHIEVE($g$)
    $s_0^{\mathsf{B}} \leftarrow s_0[\![\pi]\!][\mathcal{V}^{\mathsf{B}}]$
    $s_\star^{\mathsf{B}} \leftarrow g$
    $O^{\mathsf{B}} \leftarrow \{o^{\mathsf{B}} \mid o \in O, \mathit{pre}(o) \subseteq R \cup B, \bigcup_{\langle c,v,\vartheta\rangle \in \mathit{effs}(o)[\mathcal{V}^{\mathsf{B}}]} c \subseteq R \cup B,$
            $\mathit{pre}(o^{\mathsf{B}}) = \mathit{pre}(o)[\mathcal{V}^{\mathsf{B}}],$
            $\mathit{effs}(o^{\mathsf{B}}) = \{\langle c[\mathcal{V}^{\mathsf{B}}], v, \vartheta\rangle \mid \langle c, v, \vartheta\rangle \in \mathit{effs}(o)[\mathcal{V}^{\mathsf{B}}]\}\}$

    $\langle o_1'^{\mathsf{B}}, \dots, o_k'^{\mathsf{B}}\rangle \leftarrow$ an FDR plan for $\Pi^{\mathsf{B}} = \langle \mathcal{V}^{\mathsf{B}}, O^{\mathsf{B}}, s_0^{\mathsf{B}}, s_\star^{\mathsf{B}}\rangle$
    return $\langle o_1', \dots, o_k'\rangle$
```

Figure 2: Red-black planning algorithm.

conditional effects. Figure 2 shows the adaptation of Katz and Hoffmann's pseudo-code to conditional effects. The original algorithm assumes as input the set R^+ of preconditions and goals on red variables in a fully delete-relaxed plan, i. e., $R^+ = s_\star[\mathcal{V}^{\mathsf{R}}] \cup \bigcup_{o \in \pi^+} \mathit{pre}(o)[\mathcal{V}^{\mathsf{R}}]$ where π^+ is a relaxed plan for Π. The adaptation also requires to collect the conditions of the effects that *fire* in the relaxed plan, and therefore R^+ will also include those conditions.

The original algorithm then successively selected achieving actions for R^+, until all these red facts are true. Our algorithm, however, in the presence of conditional effects, instead of selecting an action, is required to select a particular effect of an action. Then, the black condition of that effect should be achieved together with action's black precondition. Throughout the algorithm, *R denotes the set of red facts already achieved by the current red-black plan prefix π; B denotes the set of black variable values that can be achieved using only red outside conditions from R.*

To explain how the algorithm works, for each selected achieving action and effect pair (or for the goal), if the black variable values do not match the precondition of the selected

		multiply out		native support	
Domain	#	Constr.	Inv.	Constr.	Inv.
briefcaseworld	50	10	10	50	50
cavediving-14-adl	20	20	20	20	20
citycar-sat14-adl	20	20	0	20	0
fsc-blocks	14	0	0	14	0
fsc-grid-a1	16	1	0	16	0
fsc-grid-a2	2	1	0	2	0
fsc-grid-r	16	0	0	16	0
fsc-hall	2	1	0	2	0
fsc-visualmarker	7	0	0	7	0
gedp-ds2ndp	24	4	4	24	0
miconic-simple	150	150	150	150	150
t0-adder	2	0	0	2	2
t0-coins	30	20	15	30	30
t0-comm	25	25	0	25	0
t0-grid-dispose	15	0	0	15	15
t0-grid-push	5	0	0	5	5
t0-grid-trash	1	0	0	1	1
t0-sortnet	5	0	0	5	0
t0-sortnet-alt	6	1	0	6	0
t0-uts	29	13	0	29	4
Sum	439	266	199	439	277

Table 1: Per-domain summary of the number of tasks (#); number of constructed (Constr.) and number of tasks with invertible variables (Inv.) for both approaches.

action and effect condition (or the goal), the algorithm attempts to achieve the aforementioned partial state. For that, ACHIEVE(g) is called, which finds a sequence of actions achieving the partial state by solving the black subtask Π^{B} with invertible variables.

Katz and Hoffmann (2014b) present a simple algorithm that solves the black subtask: Starting at the leaf variables and working up to the roots, the plan is constructed by augmenting the partial plan with plan fragments that correspond to paths in domain transition graphs, bringing the supporting variables into place [1]. They show the algorithm runtime to be polynomial in the size of Π^{B} and the length of the plan returned, which, although worst-case exponential in the size of Π^{B}, is practically efficient. As conditional effects correspond to edges in domain transition graphs, the algorithm works verbatim for tasks with conditional effects.

Experimental Evaluation

In order to evaluate the benefit of natively support conditional effects in red-black planning heuristics, we implemented the support within the existing implementation of red-black planning heuristics, adapted to the current Fast Downward framework (Helmert 2006). We compared our native support (NS) to the original implementation on top of a transformation that multiplies out conditional effects (MO), a baseline for our comparison. We perform a greedy best first search with a single queue, ordered by the red-black

[1]A similar algorithm was mentioned, but not used, by Helmert Helmert (2006)

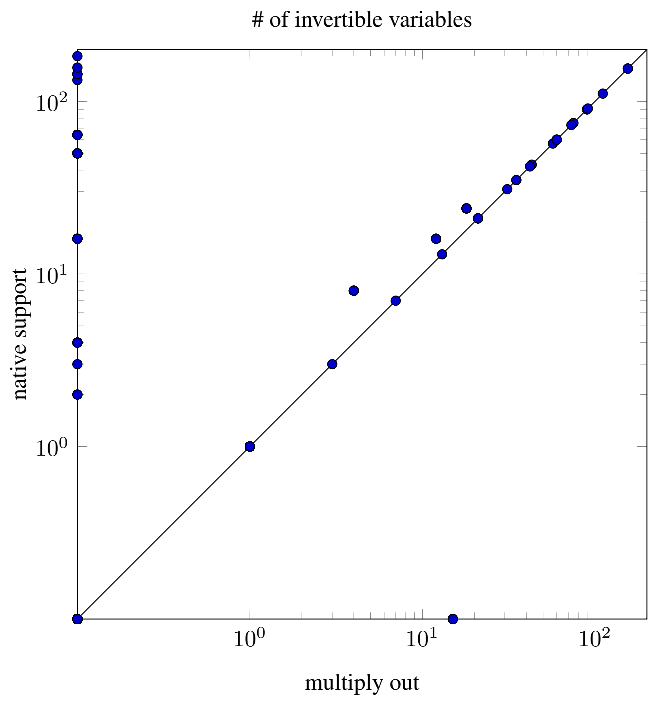


Figure 3: Comparison of the number of RSE-invertible variables with the baseline multiplying out conditional effects and the native support for conditional effects.

heuristic. The heuristic is obtained by solving a red-black planning task with black DAG causal graph. The red-black planning task is obtained by coloring the RSE-invertible variables black as long as the black part is a DAG (Domshlak, Hoffmann, and Katz 2015).

The experiments were performed on Intel(R) Xeon(R) CPU E7-8837 @2.67GHz machines, with the time and memory limit of 30min and 2GB, respectively. We performed our evaluation on the existing set of benchmark domains with conditional effects. There are only a handful number of domains from previous International Planning Competitions (IPC) with conditional effects and no axioms. Therefore, following Haslum (2013), we also use problems generated by the conformat-to-classical planning compilation (T0) (Palacios and Geffner 2009) and the finite-state controller synthesis compilation (FSC) (Bonet, Palacios, and Geffner 2009). In addition, following Röger, Pommerening, and Helmert (2014), we use the briefcase world domain from the IPP benchmark collection (Köhler 1999) and the Miconic simpleadl version from the benchmark set of the International Planning Competition (IPC2000), as it has conditional effects but no derived predicates after grounding with Fast Downward. This results in total of 20 domains, shown in Table 1. The table also depicts the number of tasks in these domains, as well as the per-domain summary of the number of tasks that could be constructed and the number of tasks where at least one RSE-invertible variable was found. Note that in all cases when the tasks that could not be constructed, it was due to the transformation that multiplied out

Domain	#	Inv.	**multiply out**	**native support**
briefcaseworld	50	50	10	**50**
cavediving-14-adl	20	20	**7**	**7**
gedp-ds2ndp	24	4	0	**4**
miconic-simple	150	150	**150**	**150**
t0-adder	2	2	0	0
t0-coins	30	30	**20**	**20**
t0-grid-dispose	15	15	0	**15**
t0-grid-push	5	5	0	**3**
t0-grid-trash	1	1	0	0
t0-uts	29	4	**4**	3
Sum	326	281	191	**252**

Table 2: Per-domain coverage for domains with invertible variables, tasks for which at least one approach found invertible variables. Best performers are marked in bold.

conditional effects created a task that was too large to fit into memory. This happened in 173 out of the total 439 tasks. Note that there is no way around creating the transformed task in order to check RSE-invertibility of variables in the transformed task. If no RSE-invertible variables are found, red-black heuristics are effectively equivalent to a FF heuristic (Hoffmann and Nebel 2001). Importantly, in such cases, for the baseline approach FF heuristic would be constructed on top of the transformed task. Looking at the table, there are 10 domains where either of the two approached found any RSE-invertible variables. In what follows, we restrict our attention to these domains.

Figure 3 compares the number of RSE-invertible variables for each task in these 10 domains. Each point corresponds to a task, showing the number of RSE-invertible variables for the multiply-out approach and for the native one. Observe that the number of RSE-invertible variables mostly increases when moving from compiling away conditional effects to natively supporting them. The points on “multiply out” axis corresponds to 4 tasks in gedp-ds2ndp domain, where the multiply-out transformation resulted in a tasks with 15 RSE-invertible variables, while the original task has 0 RSE-invertible variable under our new definition of RSE-invertibility in the presence of conditional effect. For the reverse case (“native support” axis), there are 82 tasks where there could not be found any RSE-invertible variables in the transformed task, including due to failing to create the task, and there was at least one RSE-invertible variable under our new definition, allowing us to run the red-black planning heuristic.

Turning our attention to the heuristics performance, Table 2 shows the per-domain coverage, comparing our suggested approach to the baseline. The second column describes the overall number of instances in each domain, while the third column shows the number of instances on which at least one of the approaches found invertible variables (Inv.). The last two colums report the number of solved instances for the compared approaches, restricting to the set of instances in Inv., as described above. The table clearly shows the benefit of handling conditional effects natively in the heuristic – the coverage reduces by 1 instance in one domain,

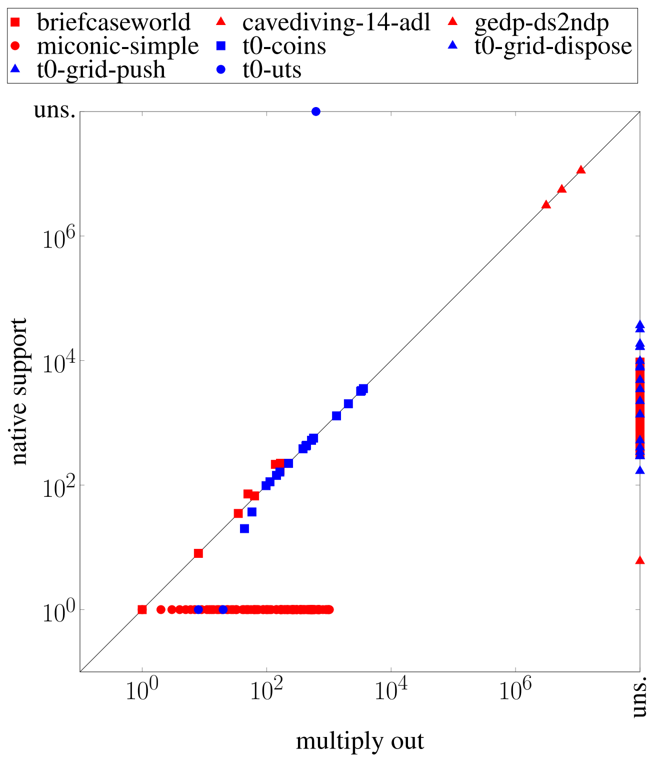


Figure 4: Domain-wise comparison of the number of evaluations performed with the baseline multiplying out conditional effects and the native support for conditional effects.

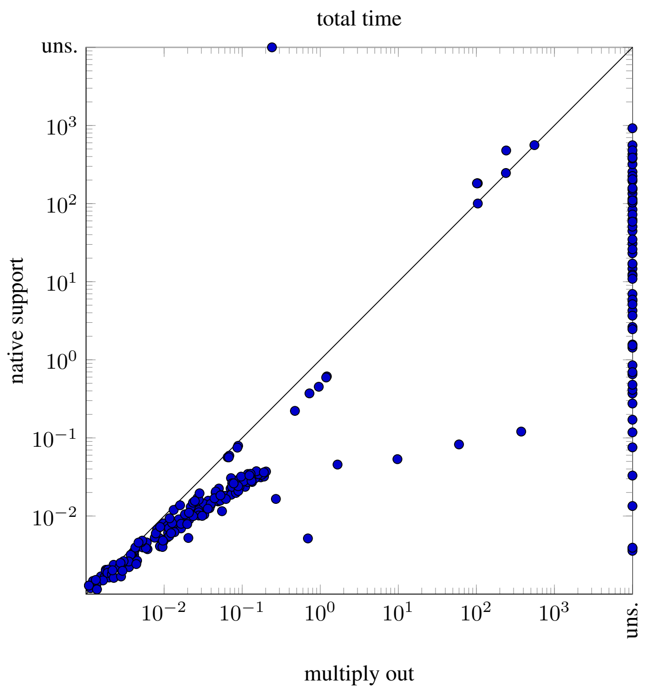


Figure 5: Comparison of the total time untill a solution is found with the baseline multiplying out conditional effects and the native support for conditional effects.

remains the same in five domains, and increases by 62 instances in four domains, namely by 40, 15, 4, and 3 in briefcaseworld, t0-grid-dispose, gedp-ds2ndp, and t0-grid-push, respectively. Note that if we compare the coverage on all instances of all 20 domains, the baseline solves 246 out of 439 instances, while our approach solves 359. However, on tasks where no RSE-invertible variables were found, such comparison would essentially be between the FF heuristics with and without task transformation.

In order to show a per-instance comparison, Figure 4 compares the number of heuristic evaluations performed until the solution is found. Here as well, we restrict our attention to tasks with RSE-invertible variables. First, observe that many tasks previously not solved, are now solved after up to 10000 evaluations, with a few (five) additional examples going as high as 37000 evaluations. Second, in many more cases the task is solved without search at all, after 1 evaluation – all instances in miconic domain and 3 instances in t0-uts. There are only a few instances where the preformance got negligebly worse, and many where it improved considerably.

Finally, to measure the heuristic computation time difference, Figure 5 shows the per-instance total time comparison. Here as well, the picture is clear. The vast majority of points are under the diagonal, demonstrating our conjecture that it pays off in practice to support conditional effects natively in red-black planning heuristics.

Conclusions and Future Work

We have shown how to adapt red-black planning to support conditional effects, an important modeling feature in planning. To that end, we have adapted the formalism of red-black planning to include conditional effects and have shown how to derive practical red-black planning heuristic. For that, we extended the definition of invertibility of variables and adapted the existing algorithms for tractable red-black planning for the classical formalism to work in the presence of conditional effects. To measure the benefit of natively supporting conditional effects in the red-black planning heuristic, we performed an extensive experimental evaluation, comparing to the red-black planning heuristic without the support for conditional effects, applied to a transformed planning task, with conditional effects compiled away. Our evaluation clearly shows the benefit of supporting conditional effects natively.

For future work, we intend to explore natively supporting additional non-classical features in red-black planning, such as axioms and derived predicates (McDermott et al. 1998; Thiébaux, Hoffmann, and Nebel 2005). Further, an interesting direction is extending red-black planning to richer formalisms, such as fully observable nondeterministic (FOND) planning.

References

Baier, J. A., and Botea, A. 2009. Improving planning performance using low-conflict relaxed plans. In Gerevini, A.; Howe, A.; Cesta, A.; and Refanidis, I., eds., *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*, 10–17. AAAI Press.

Bonet, B., and Geffner, H. 2001. Planning as heuristic search. *Artificial Intelligence* 129(1):5–33.

Bonet, B.; Palacios, H.; and Geffner, H. 2009. Automatic derivation of memoryless policies and finite-state controllers using classical planners. In Gerevini, A.; Howe, A.; Cesta, A.; and Refanidis, I., eds., *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*, 34–41. AAAI Press.

Cai, D.; Hoffmann, J.; and Helmert, M. 2009. Enhancing the context-enhanced additive heuristic with precedence constraints. In Gerevini, A.; Howe, A.; Cesta, A.; and Refanidis, I., eds., *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*, 50–57. AAAI Press.

Chen, H., and Giménez, O. 2010. Causal graphs and structurally restricted planning. *Journal of Computer and System Sciences* 76(7):579–592.

Domshlak, C.; Hoffmann, J.; and Katz, M. 2015. Red-black planning: A new systematic approach to partial delete relaxation. *Artificial Intelligence* 221:73–114.

Fox, M., and Long, D. 2001. Stan4: A hybrid planning strategy based on subproblem abstraction. *AI Magazine* 22(3):81–84.

Gerevini, A.; Saetti, A.; and Serina, I. 2003. Planning through stochastic local search and temporal action graphs in LPG. *Journal of Artificial Intelligence Research* 20:239–290.

Haslum, P. 2012. Incremental lower bounds for additive cost planning problems. In McCluskey, L.; Williams, B.; Silva, J. R.; and Bonet, B., eds., *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling (ICAPS 2012)*, 74–82. AAAI Press.

Haslum, P. 2013. Optimal delete-relaxed (and semi-relaxed) planning with conditional effects. In Rossi, F., ed., *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, 2291–2297. AAAI Press.

Helmert, M., and Geffner, H. 2008. Unifying the causal graph and additive heuristics. In Rintanen, J.; Nebel, B.; Beck, J. C.; and Hansen, E., eds., *Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS 2008)*, 140–147. AAAI Press.

Helmert, M. 2004. A planning heuristic based on causal graph analysis. In Zilberstein, S.; Koehler, J.; and Koenig, S., eds., *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004)*, 161–170. AAAI Press.

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

Hoffmann, J., and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research* 14:253–302.

Katz, M., and Hoffmann, J. 2013. Red-black relaxed plan heuristics reloaded. In Helmert, M., and Röger, G., eds., *Proceedings of the Sixth Annual Symposium on Combinatorial Search (SoCS 2013)*, 105–113. AAAI Press.

Katz, M., and Hoffmann, J. 2014a. Mercury planner: Pushing the limits of partial delete relaxation. In *Eighth International Planning Competition (IPC-8): planner abstracts*, 43–47.

Katz, M., and Hoffmann, J. 2014b. Pushing the limits of partial delete relaxation: Red-black DAG heuristics. In *ICAPS 2014 Workshop on Heuristics and Search for Domain-independent Planning (HSDIP)*, 40–44.

Katz, M.; Hoffmann, J.; and Domshlak, C. 2013a. Red-black relaxed plan heuristics. In desJardins, M., and Littman, M. L., eds., *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI 2013)*, 489–495. AAAI Press.

Katz, M.; Hoffmann, J.; and Domshlak, C. 2013b. Who said we need to relax *all* variables? In Borrajo, D.; Kambhampati, S.; Oddi, A.; and Fratini, S., eds., *Proceedings of the Twenty-Third International Conference on Automated Planning and Scheduling (ICAPS 2013)*, 126–134. AAAI Press.

Keyder, E.; Hoffmann, J.; and Haslum, P. 2012. Semi-relaxed plan heuristics. In McCluskey, L.; Williams, B.; Silva, J. R.; and Bonet, B., eds., *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling (ICAPS 2012)*, 128–136. AAAI Press.

Köhler, J. 1999. Handling of conditional effects and negative goals in IPP. Technical Report 128, University of Freiburg, Department of Computer Science.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL – The Planning Domain Definition Language – Version 1.2. Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, Yale University.

Nebel, B. 2000. On the compilability and expressive power of propositional planning formalisms. *Journal of Artificial Intelligence Research* 12:271–315.

Palacios, H., and Geffner, H. 2009. Compiling uncertainty away in conformant planning problems with bounded width. *Journal of Artificial Intelligence Research* 35:623–675.

Richter, S., and Westphal, M. 2010. The LAMA planner: Guiding cost-based anytime planning with landmarks. *Journal of Artificial Intelligence Research* 39:127–177.

Röger, G.; Pommerening, F.; and Helmert, M. 2014. Optimal planning in the presence of conditional effects: Extending LM-Cut with context splitting. In Schaub, T.; Friedrich, G.; and O'Sullivan, B., eds., *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014)*, 765–770. IOS Press.

Thiébaux, S.; Hoffmann, J.; and Nebel, B. 2005. In defense of PDDL axioms. *Artificial Intelligence* 168(1–2):38–69.