AI Planning

10. Partial Delete Relaxation

How to (Systematically!) Take Some Delete Effects Into Account

Álvaro Torralba, Cosmina Croitoru

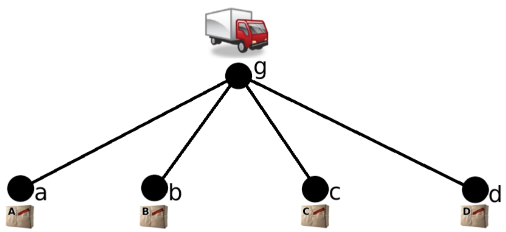
SAARLAND UNIVERSITY
COMPUTER SCIENCE

Winter Term 2018/2019

Thanks to Prof. Jörg Hoffmann for slide sources

Agenda

Take This, h^+! "Star-Shape Logistics"



- State variables: $v_T : \{g, a, b, c, d\}$; $v_A, v_B, v_C, v_D : \{t, g, a, b, c, d\}$.
- Initial state: $v_T = g, v_A = a, v_B = b, v_C = c, v_D = d$.
- Goal: $v_A = g, v_B = g, v_C = g, v_D = g$.
- Actions (unit costs): $drive(x, y), load(x, y), unload(x, y)$.
  E.g., $load(x, y)$ has precondition $v_T = y, v_x = y$ and effect $v_x = t$.

→ Relaxed plan for this task: $drive(g, a)$, $drive(g, b)$, $drive(g, c)$, $drive(g, d)$, $load(A, a)$, $load(B, b)$, $load(C, c)$, $load(D, d)$, $unload(A, g)$, $unload(B, g)$, $unload(C, g)$, $unload(D, g)$. Thus: $h^+ = 12 < 16 = h^*$.

→ And with 100 star-leaf locations & packages? $h^+ = 300 \ll 400 = h^*$.

Quo Vadis, h^+?

Major weaknesses of the delete relaxation:

- Completely unable to account for "to-and-fro" (cf. previous slide).
- Completely unable to account for "harmful side effects" (such as fuel consumption as a side effect of driving a truck, cf. "fill up on gas once, keep driving forever ...").

"Taking some deletes into account":

- h^+: Extreme case were no deletes are taken into account. (Fast approximations, but has the weaknesses above.)
- h^*: Extreme case were all deletes are taken into account. (Perfect, but computing it would entail solving the task in the first place.)
- Partial delete relaxation interpolates between these extremes, to obtain a fast *and* good heuristic.
  → "Interpolate" = Ability to scale smoothly from h^+ all the way to h^*.
- Challenge since 2001, first achieved in 2012 (!)

Our Agenda for This Chapter

2. **Red-Black Planning:** Introduces the most recent and, arguably, most natural idea for interpolating between h^+ and h^*: Relax only some of the FDR state variables.
3. **(A Brief Glimpse of) The Complexity of Red-Black Planning:** How many state variables do we need to relax for the heuristic computation to become tractable?
4. **Red-Black Plan Heuristics in Practice:** Naïve approaches exhibit severe over-approximation. Here's how to do better.
5. **Other Methods:** A brief glimpse at the two other known partial delete relaxation methods.

Red-Black Planning

→ Black variables switch between values ("real semantics)", red variables accumulate them ("relaxed semantics").

Definition (Red-Black Planning). *A red-black planning task is a tuple $\Pi^{\mathsf{RB}} = (V^{\mathsf{B}}, V^{\mathsf{R}}, A, c, I, G)$ where V^{B} is a set of black variables, V^{R} is a set of red variables, and everything else is exactly as for FDR tasks. The semantics is:*

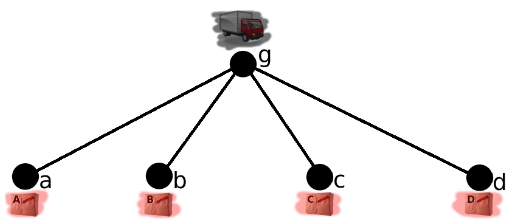
- *A state s assigns each $v \in V^{\mathsf{B}} \cup V^{\mathsf{R}}$ a subset $s(v) \subseteq D_v$, where $|s(v)| = 1$ for all $v \in V^{\mathsf{B}}$.*
- *Action a is applicable in s if $pre_a(v) \in s(v)$ for all v s.t. $pre_a(v)$ is defined.*
- *Applying a in s changes the value of black effect variables v to $\{eff_a(v)\}$, and changes the value of red effect variables v to $s(v) \cup \{eff_a(v)\}$.*
- *A state s is a goal state if $G[v] \in s(v)$ for all v s.t. $G(v)$ is defined.*

Given an FDR task $\Pi = (V, A, c, I, G)$ and a subset $V^{\mathsf{R}} \subseteq V$ of variables, the red-black relaxation of Π is the red-black task $\Pi^{\mathsf{RB}}_{V^{\mathsf{R}}} = (V \setminus V^{\mathsf{R}}, V^{\mathsf{R}}, A, c, I, G)$. A plan for Π^{RB} is a red-black plan for Π.

Notation: $h^{*\mathsf{RB}} : S \mapsto \mathbb{R}^+_0 \cup \{\infty\}$ is the cost of an optimal red-black plan for s.

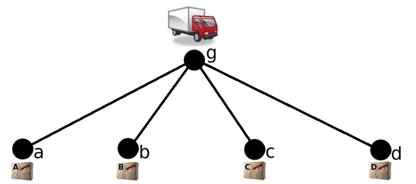
Red-Black Planning in Star-Shape Logistics

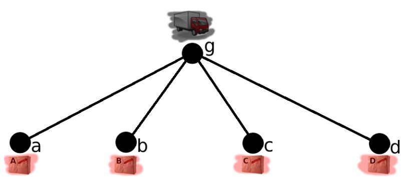
Idea: The truck moves to-and-fro, so h^+ loses information with respect to variable v_T. Let's see what happens when we paint v_T black.



- Black State variables: $v_T : \{g, a, b, c, d\}$.
- Red State variables: $v_A, v_B, v_C, v_D : \{t, g, a, b, c, d\}$.
- Initial state: $v_T = g, v_A = a, v_B = b, v_C = c, v_D = d$.
- Goal: $v_A = g, v_B = g, v_C = g, v_D = g$.
- Actions (unit costs): $drive(x, y), load(x, y), unload(x, y)$.
  E.g., $load(x, y)$ has precondition $v_T = y, v_x = y$ and effect $v_x = t$.

Red-Black Planning in Star-Shape Logistics, ctd.





Relaxed plan:

1. Initial state: $\{v_T = g, \dots\}$.
2. Apply $drive(g, a)$: $\{v_T = g, v_T = a, \dots\}$.
3. Apply $drive(g, b)$: $\{v_T = g, v_T = a, v_T = b, \dots\}$.
4. ...

Red-black plan:

1. Initial state: $\{v_T = g, \dots\}$.
2. Apply $drive(g, a)$: $\{v_T = a, \dots\}$.
3. Apply $drive(g, b)$: Not applicable!

→ It's easy to see that any optimal red-black plan is a real plan here. In particular, $h^{*\mathsf{RB}}(I) = h^*(I)$.

Basic Observations About Red-Black Planning

Reminder: Given an FDR task (V, A, c, I, G) and a subset $V^{\mathsf{R}} \subseteq V$ of variables, the red-black relaxation of Π is $(V \setminus V^{\mathsf{R}}, V^{\mathsf{R}}, A, c, I, G)$.

- If we set $V^{\mathsf{R}} := V$, then $h^{*\mathsf{RB}} = h^+$.
- If we set $V^{\mathsf{R}} := \emptyset$, then $h^{*\mathsf{RB}} = h^*$.

→ Red-black planning allows to naturally interpolate between h^+ and h^*.

→ So, that's it? In our planner, we'll set $V^{\mathsf{R}} := \emptyset$ and be done? Nope: Computing $h^{*\mathsf{RB}}$ would just mean to solve the original planning task.

→ Choosing V^{R} = Trading off between accuracy and overhead.

→ *How many variables do we have to paint red in order to obtain a tractable (polynomial-time solvable) red-black planning problem?*

Questionnaire

Question!

What if, in Star-Shape Logistics, instead of the truck we paint the packages black?

(A): $h^{*\mathsf{RB}} = h^*$

(B): $h^{*\mathsf{RB}} = h^+$

(C): We can't paint the packages black

(D): Honestly, I don't care what color the packages have

→ (A): No, because painting the packages black has no effect at all on the relaxed plan. The packages do not "move to-and-fro" anyway, each just makes two transitions to its goal value.

→ (B): Yes, see (A).

→ (C): We can paint whatever variable subset we want.

→ (D): In fact, it doesn't matter (to the heuristic value) what color the packages have: see (A). And that's actually the case for *any* causal graph leaf variables, which are "pure clients" and don't need to move to-and-fro (cf. **Chapter 5**, see [Katz *et al.* (2013b)] for details).

"How Many Variables do We Have to Paint Red" = All??

Theorem (Hardness for a Single Black Variable). *The problem of deciding, given a red-black planning task $\Pi^{\mathsf{RB}} = (V^{\mathsf{B}}, V^{\mathsf{R}}, A, c, I, G)$ where $|V^{\mathsf{B}}| = 1$, whether Π^{RB} is solvable, is* **NP**-*complete.*

Proof Sketch. (Membership: Omitted) Hardness: By reduction from SAT.

- Red variables: For each variable $v_i \in \{v_1, \dots, v_m\}$ in the CNF, a variable v_i with domain $D_{v_1} = \{none, true, false\}$: Has v_i been assigned yet? And to which value? Initially $v_i = none$.
  For each clause $c_j \in \{c_1, \dots, c_n\}$ in the CNF, a Boolean variable sat_j: Has clause j been satisfied yet? Initially, sat_j is false; the goal requires it to be true.
- Black variable: v_0 with domain $D_{v_0} = \{1, \dots, n+1\}$: Whose variable's turn is it to be assigned? Initially, $v_0 = 1$.
- Actions that allow setting v_i from $none$ to either $true$ or $false$, provided that $v_0 = i$; apart from setting v_i, the actions also set $v_0 := i + 1$.
- Actions that allow to make sat_j true provided one of its literals has already been assigned to the correct truth value.

→ We cannot "cheat" because the black "index variable" v_0 forces us to assign each v_i exactly once!

Simple Structure, Part I: The Black Causal Graph

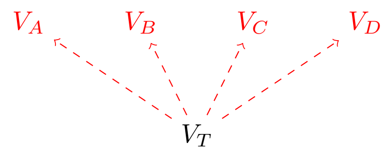
The theorem holds for worst-case structure of the black variables.

→ To the rescue: Choose the red variables so that the structure of the black variables is "simple"!

Definition (Black Causal Graph). *Let $\Pi^{\mathsf{RB}} = (V^{\mathsf{B}}, V^{\mathsf{R}}, A, c, I, G)$ be a red-black planning task. The black causal graph of Π^{RB} is the directed graph with vertices V^{B} and an arc (u, v) whenever there exists an action $a \in A$ so that either (i) there exists $a \in A$ so that $pre_a(u)$ and $eff_a(v)$ are both defined, or (ii) there exists $a \in A$ so that $eff_a(u)$ and $eff_a(v)$ are both defined.*

→ The subgraph of the causal graph induced by the black variables.

→ The black causal graph in Star-Shape Logistics:



→ Relevant for us here: There are no arcs between black variables.

Simple Structure, Part II: Invertible Variables

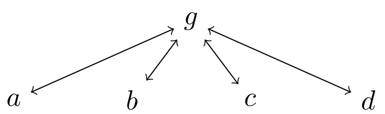
Reminder: → Chapter 5

Let $\Pi = (V, A, c, I, G)$ be an FDR planning task, and let $v \in V$. The domain transition graph (DTG) of v is the arc-labeled directed graph with vertices D_v, and, for every $d, d' \in D_v$ and $a \in A$ where either (i) $pre_a(v) = d$ and $eff_a(v) = d'$ or (ii) $pre_a(v)$ is not defined and $eff_a(v) = d'$, an arc $d \xrightarrow{a} d'$.
We refer to $d \xrightarrow{a} d'$ as a value transition of v. We write $d \xrightarrow{a}_{\varphi} d'$ where $\varphi = pre_a \setminus \{v = d\}$ is the outside condition.

Let $d \rightarrow_{\varphi} d'$ be a value transition of v. We say that $d \rightarrow_{\varphi} d'$ is invertible if there exists a value transition $d' \rightarrow_{\varphi'} d$ where $\varphi' \subseteq \varphi$.

Notation: A variable is invertible if all transitions in its DTG are invertible.

→ The DTG of the truck variable v_T in Star-Shape Logistics:



→ Relevant for us here: v_T is invertible.

The SMS Theorem

Theorem ("The SMS Theorem"). *Let $\Pi = (V, A, c, I, G)$ be an FDR planning task, and let $V^{\mathsf{R}} \subseteq V$ be a subset of its state variables. Say that, in the red-black relaxation of Π, the black causal graph does not contain any arcs, and all black variables are invertible. Then any relaxed plan for Π can in polynomial time be transformed into a red-black plan for Π.*

- **Idea:** Relaxed Plan Repair. Execute the relaxed plan step-by-step. If a black precondition (or goal) is not satisfied, we can move each black variable concerned into its required precondition/goal value separately.
- **Corollary (a):** If a relaxed plan exists, we can easily generate a red-black plan. **Trivial (b):** If no relaxed plan exists, then no red-black plan can exist either. From (a) + (b), we have a complete and efficient red-black planning procedure.
- **Usage:** On any state s encountered during search, generate a red-black plan for s and take its cost as the heuristic value. (= "In h^{FF}, replace relaxed plan by red-black plan.")

Relaxed Plan Repair: Idea

By the SMS Theorem's prerequisites:

(a) Every black variable is invertible. E.g., truck can always move back directly.
(b) Every action moves at most one black variable.
(c) If a moves a black variable v, all outside conditions on $v' \neq v$ are red.
E.g., $drive(x, y)$ has precondition $v_T = x$ and effect $v_T = y$.
E.g., if we paint the truck red and the packages black, $load(x, y)$ has precondition $v_T = y, v_x = y$ and effect $v_x = t$.

Relaxed plan repair algorithm: Assume relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$

- $s :=$ red-black outcome of a_1 in initial state.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
  → (a) Path exists, as v is invertible: Go back to $I(v)$, then follow $\vec{a}^+$ to z.
  → (b) Moving v does not affect any other black variables.
  → (c) All outside conditions used by the path are red; and have already been achieved during our execution so far, thus they are true.
- $s :=$ red-black outcome of a_2. Proceed with $\langle a_3, \dots, a_n \rangle$ and the goal.

Relaxed Plan Repair: Pseudo-Code

// $\Pi = (V, A, c, I, G)$, relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$, black and red variables V^{B}, V^{R}
$\vec{a} := \langle a_1 \rangle$; $s := I[\![a_1]\!]$ *// red-black semantics (slide 8)*
for $i = 2$ to n **do** *// Repair black action preconditions*
  if $pre_{a_i}(V^{\mathsf{B}}) \not\subseteq s$ **then**
    $\vec{a}^{\mathsf{B}} :=$ Achieve$(s, pre_{a_i}(V^{\mathsf{B}}))$; $\vec{a} := \vec{a} \circ \vec{a}^{\mathsf{B}}$; $s := s[\![\vec{a}^{\mathsf{B}}]\!]$
  endif
  $\vec{a} := \vec{a} \circ \langle a_i \rangle$; $s := s[\![a_i]\!]$
endfor
if $G(V^{\mathsf{B}}) \not\subseteq s$ **then** *// Repair black goals*
  $\vec{a}^{\mathsf{B}} :=$ Achieve$(s, G(V^{\mathsf{B}}))$; $\vec{a} := \vec{a} \circ \vec{a}^{\mathsf{B}}$
endif
return $\vec{a}$

Procedure: Achieve(s, g)
$\vec{a}^{\mathsf{B}} := \langle\rangle$
for $v \in V^{\mathsf{B}}$ s.t. $g(v)$ is defined **do** *// Move black variables into place separately*
  $\vec{a}^{\mathsf{B}} := \vec{a}^{\mathsf{B}} \circ$ invert path used by $\vec{a}$ from $I(v)$ to $s(v)$
  $\vec{a}^{\mathsf{B}} := \vec{a}^{\mathsf{B}} \circ$ path used by $\vec{a}^+$ from $I(v)$ to $g(v)$
endfor
return $\vec{a}^{\mathsf{B}}$

Questionnaire

Theorem ("The SMS Theorem"). *Let $\Pi = (V, A, c, I, G)$ be an FDR planning task, and let $V^{\mathrm{R}} \subseteq V$ be a subset of its state variables. Say that, in the red-black relaxation of Π, the black causal graph does not contain any arcs, and all black variables are invertible. Then any relaxed plan for Π can in polynomial time be transformed into a red-black plan for Π.*

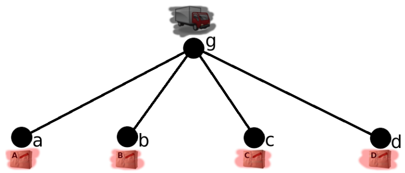
Question!

Why is this called "The SMS Theorem"?

→ After spending 3 days examining the red-black tractability borderline during a visit to Carmel Domshlak in Haifa, Jörg had this particular idea while already on the train to the airport. Based on the concepts we had already developed at the time, the proof took 3 SMS to communicate . . .

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



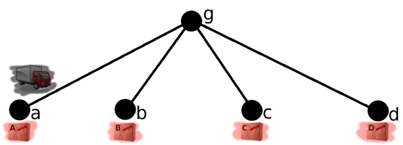
Relaxed Plan Repair:

- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $drive(g, a)$, $drive(g, b)$, $drive(g, c)$, $drive(g, d)$, $load(A, a)$, $load(B, b)$, $load(C, c)$, $load(D, d)$, $unload(A, g)$, $unload(B, g)$, $unload(C, g)$, $unload(D, g)$.

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

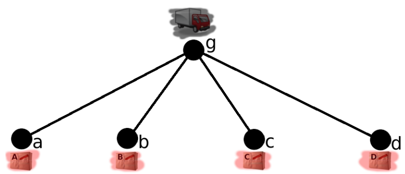
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $drive(g, a)$, $drive(g, b)$, $drive(g, c)$, $drive(g, d)$, $load(A, a)$, $load(B, b)$, $load(C, c)$, $load(D, d)$, $unload(A, g)$, $unload(B, g)$, $unload(C, g)$, $unload(D, g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

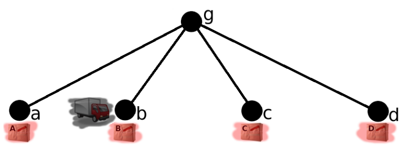
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $drive(g, b)$, $drive(g, c)$, $drive(g, d)$, $load(A, a)$, $load(B, b)$, $load(C, c)$, $load(D, d)$, $unload(A, g)$, $unload(B, g)$, $unload(C, g)$, $unload(D, g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g, b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

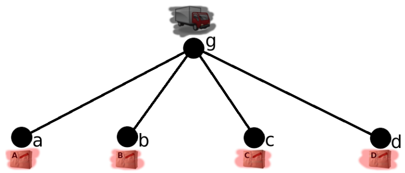
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $drive(g, b)$, $drive(g, c)$, $drive(g, d)$, $load(A, a)$, $load(B, b)$, $load(C, c)$, $load(D, d)$, $unload(A, g)$, $unload(B, g)$, $unload(C, g)$, $unload(D, g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g, b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

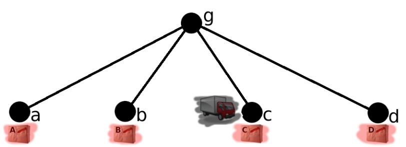
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $drive(g, c)$, $drive(g, d)$, $load(A, a)$, $load(B, b)$, $load(C, c)$, $load(D, d)$, $unload(A, g)$, $unload(B, g)$, $unload(C, g)$, $unload(D, g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g, b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$
- $a_3 = drive(g, c)$: Move v_T back to g. Apply a_3, giving $v_T = c$, $v_x = \dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

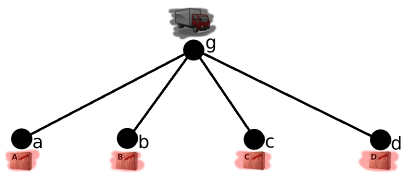
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $drive(g, c)$, $drive(g, d)$, $load(A, a)$, $load(B, b)$, $load(C, c)$, $load(D, d)$, $unload(A, g)$, $unload(B, g)$, $unload(C, g)$, $unload(D, g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g, b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$
- $a_3 = drive(g, c)$: Move v_T back to g. Apply a_3, giving $v_T = c$, $v_x = \dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

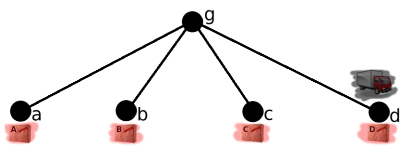
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $drive(g, d)$, $load(A, a)$, $load(B, b)$, $load(C, c)$, $load(D, d)$, $unload(A, g)$, $unload(B, g)$, $unload(C, g)$, $unload(D, g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g, b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$
- $a_3 = drive(g, c)$: Move v_T back to g. Apply a_3, giving $v_T = c$, $v_x = \dots$
- $\dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

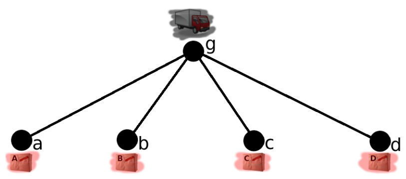
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $drive(g,d)$, $load(A,a)$, $load(B,b)$, $load(C,c)$, $load(D,d)$, $unload(A,g)$, $unload(B,g)$, $unload(C,g)$, $unload(D,g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g,b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$
- $a_3 = drive(g,c)$: Move v_T back to g. Apply a_3, giving $v_T = c$, $v_x = \dots$
- $\dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

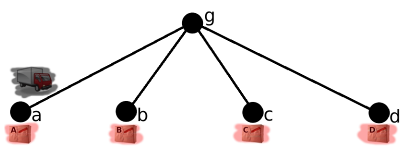
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $load(A,a)$, $load(B,b)$, $load(C,c)$, $load(D,d)$, $unload(A,g)$, $unload(B,g)$, $unload(C,g)$, $unload(D,g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g,b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$
- $a_3 = drive(g,c)$: Move v_T back to g. Apply a_3, giving $v_T = c$, $v_x = \dots$
- $\dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

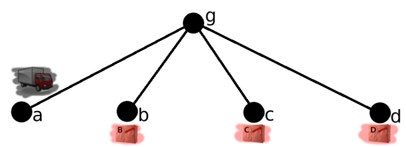
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $load(A,a)$, $load(B,b)$, $load(C,c)$, $load(D,d)$, $unload(A,g)$, $unload(B,g)$, $unload(C,g)$, $unload(D,g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g,b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$
- $a_3 = drive(g,c)$: Move v_T back to g. Apply a_3, giving $v_T = c$, $v_x = \dots$
- $\dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

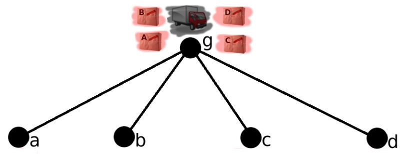
- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder: $load(A,a)$, $load(B,b)$, $load(C,c)$, $load(D,d)$, $unload(A,g)$, $unload(B,g)$, $unload(C,g)$, $unload(D,g)$.

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g,b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$
- $a_3 = drive(g,c)$: Move v_T back to g. Apply a_3, giving $v_T = c$, $v_x = \dots$
- $\dots$

Relaxed Plan Repair in Star-Shape Logistics

[**Note:** In the illustration, the packages move. This is just for simplicity of illustration: as these variables are red, actually the packages accumulate their positions.]



Relaxed Plan Repair:

- Relaxed plan $\vec{a}^+ = \langle a_1, \dots, a_n \rangle$.
- $s :=$ red-black outcome of a_1 in init.
- For any black v, if $s(v) \neq z$ precondition of a_2: Move v to value z.
- $s :=$ red-black outcome of a_2.
- Proceed with $a_3, \dots, a_n$ and the goal.

Relaxed Plan Remainder:

- After a_1 moving v_T to a: $v_T = a$, $v_x = \dots$
- $a_2 = drive(g, b)$: Move v_T back to g. Apply a_2, giving $v_T = b$, $v_x = \dots$
- $a_3 = drive(g, c)$: Move v_T back to g. Apply a_3, giving $v_T = c$, $v_x = \dots$
- ...

Questionnaire

Question!

Does Relaxed Plan Repair yield an accurate heuristic function?

(A): Yes (B): No

→ Pro: It does "take some deletes into account" and can in this way improve over standard relaxed plan heuristics.

→ Contra: It may drastically over-estimate! See previous slide: The relaxed plan schedules all truck moves up front, to the effect that the repaired red-black plan starts off by moving the truck all over the place uselessly, only to have to do it all again when the load/unload actions come up ...

How to Choose the Red Variables?

Input: A planning task $\Pi = (V, A, I, G)$

Output: Partitioning of V into V^{B} and V^{R}

Method:

1. Compute the black causal graph, and the DTG for each $v \in V$
2. Initialize $V^{\mathsf{B}} := V$ and $V^{\mathsf{R}} := \emptyset$
3. For all $v \in V^{\mathsf{B}}$: if v is not invertible then $V^{\mathsf{B}} := V^{\mathsf{B}} \setminus \{v\}$, $V^{\mathsf{R}} := V^{\mathsf{R}} \cup \{v\}$
4. While black causal graph contains arc (v, v') between $v, v' \in V^{\mathsf{B}}$ do: (*) choose $w \in \{v, v'\}$; $V^{\mathsf{B}} := V^{\mathsf{B}} \setminus \{w\}$, $V^{\mathsf{R}} := V^{\mathsf{R}} \cup \{w\}$

→ How to make the choice (*)? Prefer w that are "handled Ok by the delete relexation". (E.g.: Small number of conflicts in a relaxed plan when painting w black.)

Questionnaire

Consider the same relaxed plan: $drive(g, a)$, $drive(g, b)$, $drive(g, c)$, $drive(g, d)$, $load(A, a)$, $load(B, b)$, $load(C, c)$, $load(D, d)$, $unload(A, g)$, $unload(B, g)$, $unload(C, g)$, $unload(D, g)$.

Question!

What does Relaxed Plan Repair do if, in Star-Shape Logistics, instead of the truck we paint the packages black?

(A): Nothing (B): Same as Before

→ The black preconditions now have the form "$v_X = x$" and "$v_X = t$" where X stands for a package $\{A, B, C, D\}$, all of which are satisfied when execution arrives at the respective "$load(X, x)$" respectively "$unload(X, g)$" action. The black goals now have the form "$v_x = g$" and are satisfied at the end of the execution.

→ So Relaxed Plan Repair never invokes the "Achieve" procedure, effectively doing nothing, (A).

The Problem, and a Solution

What is the problem?

- **Relaxed Plan:** $drive(g, a), drive(g, b), drive(g, c), drive(g, d),$ $load(A, a), load(B, b), load(C, c), load(D, d),$ $unload(A, g), unload(B, g), unload(C, g), unload(D, g)$.
- The relaxed plan can (and will) schedule all truck moves first. We can't.
- In general: Commitments made by relaxed plan throw us off in red-black.

What can we do about it? Let's rely *less* on the relaxed plan!

- $R^+ := [G(V^R) \cup \bigcup_{a \in \vec{a}^+} pre_a(V^R)] \setminus I$ where $\vec{a}^+$ is a relaxed plan: The red precondition/goal values achieved along the relaxed plan.
- In the example:
  $R^+ = \{v_A = t, v_A = g, v_B = t, v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$
- **Idea:** Keep selecting actions that achieve one more fact from R^+!

  $\rightarrow$ In the example, these actions will be the loads/unloads, and the truck moves will simply be inserted as a helper for achieving their preconditions.

Relaxed Facts Following: Outline

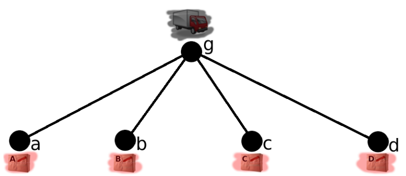
Notation:

- R: Red values already true, i.e., true in the outcome state s of the current red-black plan prefix (under red-black execution semantics).
- B: Overall set of black values $v = d$ reachable from $I(v)$ using only outside conditions from R.

Algorithm sketch:

- $s := I$. If $R \supseteq R^+$ then stop.
- Select a from $A' := \{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
  $\rightarrow$ Path exists, can be executed in s, and does not affect any other black variables: Similar arguments as for Relaxed Plan Repair.
- $s :=$ red-black outcome of a. Proceed with the rest of R^+.
- Move all black goal variables into place.
  $\rightarrow$ Possible because all of R^+, and thus all necessary outside conditions for these paths, have been achieved.

Relaxed Facts Following in Star-Shape Logistics

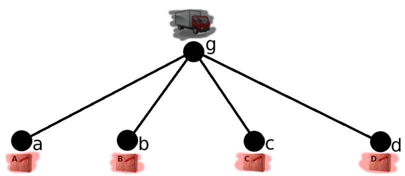


Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$
  $\rightarrow R$: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{v_A = t,\ v_A = g,\ v_B = t,\ v_B = g,\ v_C = t,\ v_C = g,\ v_D = t,\ v_D = g\}$.

Relaxed Facts Following in Star-Shape Logistics



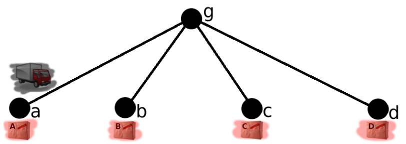
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$
  $\rightarrow R$: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{v_A = t,\ v_A = g,\ v_B = t,\ v_B = g,\ v_C = t,\ v_C = g,\ v_D = t,\ v_D = g\}$.

- $R =$init package positions; B=all truck positions.
  $\rightarrow \{load(A, a), load(B, b), load(C, c), load(D, d)\}$; select $load(A, a)$.

Relaxed Facts Following in Star-Shape Logistics



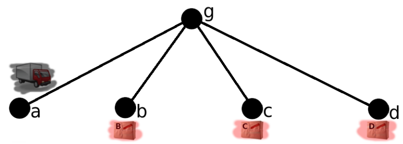
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$
  $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{v_A = t, v_A = g, v_B = t, v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$.

- $R =$init package positions; B=all truck positions.
  $\rightarrow \{load(A, a), load(B, b), load(C, c), load(D, d)\}$; select $load(A, a)$.

Relaxed Facts Following in Star-Shape Logistics



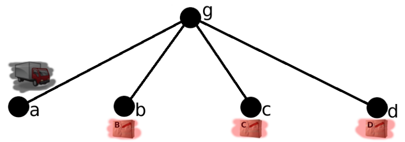
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$
  $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{v_A = t, v_A = g, v_B = t, v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$.

- $R =$init package positions; B=all truck positions.
  $\rightarrow \{load(A, a), load(B, b), load(C, c), load(D, d)\}$; select $load(A, a)$.

Relaxed Facts Following in Star-Shape Logistics



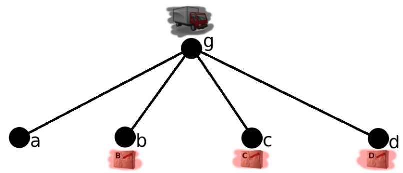
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$
  $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{ v_A = g, v_B = t, v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$.

- $R =$init package positions; B=all truck positions.
  $\rightarrow \{load(A, a), load(B, b), load(C, c), load(D, d)\}$; select $load(A, a)$.
- $R =$init package positions $+ v_A = t$; B=all truck positions.
  $\rightarrow \{unload(A, g), load(B, b), load(C, c), load(D, d)\}$; select $unload(A, g)$.

Relaxed Facts Following in Star-Shape Logistics



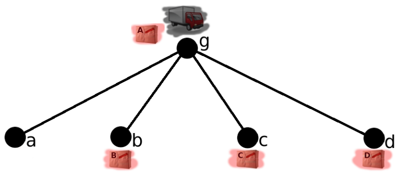
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$
  $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{ v_A = g, v_B = t, v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$.

- $R =$init package positions; B=all truck positions.
  $\rightarrow \{load(A, a), load(B, b), load(C, c), load(D, d)\}$; select $load(A, a)$.
- $R =$init package positions $+ v_A = t$; B=all truck positions.
  $\rightarrow \{unload(A, g), load(B, b), load(C, c), load(D, d)\}$; select $unload(A, g)$.

Relaxed Facts Following in Star-Shape Logistics



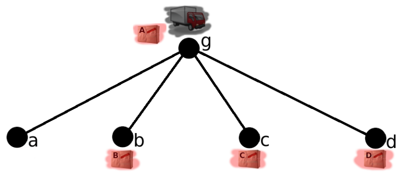
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$ $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{\ v_A = g, v_B = t, v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$.

- $R =$init package positions; B=all truck positions. $\rightarrow \{load(A, a), load(B, b), load(C, c), load(D, d)\}$; select $load(A, a)$.
- $R =$init package positions $+v_A = t$; B=all truck positions. $\rightarrow \{unload(A, g), load(B, b), load(C, c), load(D, d)\}$; select $unload(A, g)$.

Relaxed Facts Following in Star-Shape Logistics



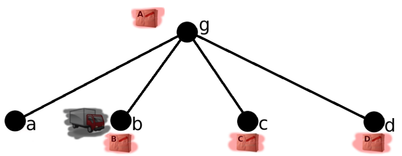
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$ $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{\ v_B = t, v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$.

- $R =$init package positions; B=all truck positions. $\rightarrow \{load(A, a), load(B, b), load(C, c), load(D, d)\}$; select $load(A, a)$.
- $R =$init package positions $+v_A = t$; B=all truck positions. $\rightarrow \{unload(A, g), load(B, b), load(C, c), load(D, d)\}$; select $unload(A, g)$.
- $R =$init package positions $+v_A = t, v_A = g$; B=all truck positions. $\rightarrow \{load(B, b), load(C, c), load(D, d)\}$; select $load(B, b)$.

Relaxed Facts Following in Star-Shape Logistics



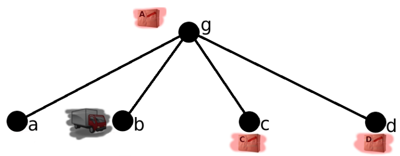
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$ $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{\ v_B = t, v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$.

- $R =$init package positions; B=all truck positions. $\rightarrow \{load(A, a), load(B, b), load(C, c), load(D, d)\}$; select $load(A, a)$.
- $R =$init package positions $+v_A = t$; B=all truck positions. $\rightarrow \{unload(A, g), load(B, b), load(C, c), load(D, d)\}$; select $unload(A, g)$.
- $R =$init package positions $+v_A = t, v_A = g$; B=all truck positions. $\rightarrow \{load(B, b), load(C, c), load(D, d)\}$; select $load(B, b)$.

Relaxed Facts Following in Star-Shape Logistics



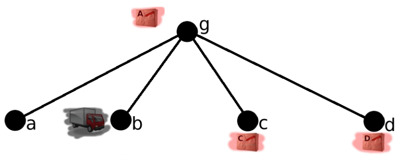
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, eff_a \cap (R^+ \setminus R) \neq \emptyset\}$ $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{\ v_B = t, v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$.

- $R =$init package positions; B=all truck positions. $\rightarrow \{load(A, a), load(B, b), load(C, c), load(D, d)\}$; select $load(A, a)$.
- $R =$init package positions $+v_A = t$; B=all truck positions. $\rightarrow \{unload(A, g), load(B, b), load(C, c), load(D, d)\}$; select $unload(A, g)$.
- $R =$init package positions $+v_A = t, v_A = g$; B=all truck positions. $\rightarrow \{load(B, b), load(C, c), load(D, d)\}$; select $load(B, b)$.

Relaxed Facts Following in Star-Shape Logistics



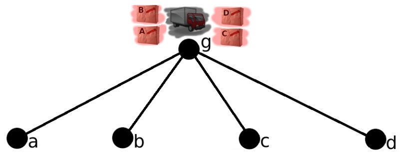
Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, \mathit{eff}_a \cap (R^+ \setminus R) \neq \emptyset\}$ $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{\ v_B = g, v_C = t, v_C = g, v_D = t, v_D = g\}$.

- $R =$init package positions; B=all truck positions. $\rightarrow \{load(A,a), load(B,b), load(C,c), load(D,d)\}$; select $load(A,a)$.
- $R =$init package positions $+v_A = t$; B=all truck positions. $\rightarrow \{unload(A,g), load(B,b), load(C,c), load(D,d)\}$; select $unload(A,g)$.
- $R =$init package positions $+v_A = t, v_A = g$; B=all truck positions. $\rightarrow \{load(B,b), load(C,c), load(D,d)\}$; select $load(B,b)$.
- ...

Relaxed Facts Following in Star-Shape Logistics



Relaxed Facts Following:

- $R^+ :=$ red values used in $\vec{a}^+$.
- $\{a \mid pre_a \subseteq R \cup B, \mathit{eff}_a \cap (R^+ \setminus R) \neq \emptyset\}$ $\rightarrow$ R: red values already true; B: black values reachable using R.
- For any black v, if $s(v) \neq z$ precondition of a: Move v to value z.
- $s :=$ red-black outcome of a. Proceed with rest of R^+, and the goal.

$R^+ = \{\quad\}$.

- $R =$init package positions; B=all truck positions. $\rightarrow \{load(A,a), load(B,b), load(C,c), load(D,d)\}$; select $load(A,a)$.
- $R =$init package positions $+v_A = t$; B=all truck positions. $\rightarrow \{unload(A,g), load(B,b), load(C,c), load(D,d)\}$; select $unload(A,g)$.
- $R =$init package positions $+v_A = t, v_A = g$; B=all truck positions. $\rightarrow \{load(B,b), load(C,c), load(D,d)\}$; select $load(B,b)$.
- ...

Relaxed Facts Following: Pseudo-Code

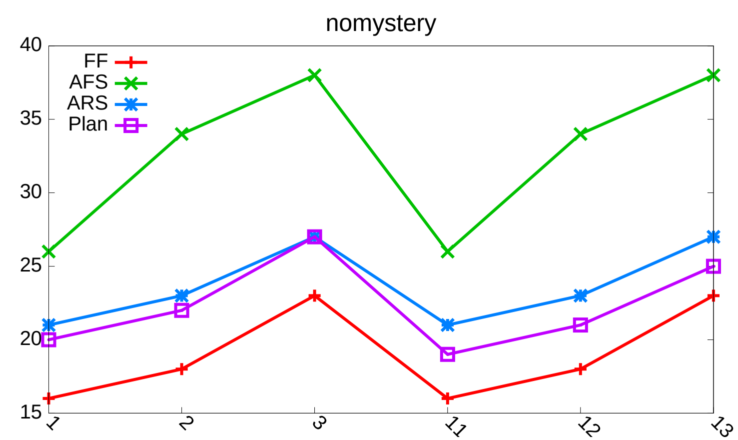
$\vec{a} := \langle\rangle$; $s := I$; UpdateRB()
while $R \not\supseteq R^+$ **do** *// Achieve one more R^+ fact*
  $A' := \{a \mid pre_a \subseteq R \cup B, \mathit{eff}_a \cap (R^+ \setminus R) \neq \emptyset\}$
  Select $a \in A'$
  if $pre_a(V^{\mathsf{B}}) \not\subseteq s$ **then**
    $\vec{a}^{\mathsf{B}} :=$ Achieve$(s, pre_a(V^{\mathsf{B}}))$; $\vec{a} := \vec{a} \circ \vec{a}^{\mathsf{B}}$; $s := s[\![\vec{a}^{\mathsf{B}}]\!]$ *// red-black semantics*
  endif
  $\vec{a} := \vec{a} \circ \langle a \rangle$; $s := s[\![a]\!]$; UpdateRB()
endwhile
if $G(V^{\mathsf{B}}) \not\subseteq s$ **then** *// Repair black goals*
  $\vec{a}^{\mathsf{B}} :=$ Achieve$(s, G(V^{\mathsf{B}}))$; $\vec{a} := \vec{a} \circ \vec{a}^{\mathsf{B}}$
endif
return $\vec{a}$

Procedure: UpdateRB() *// Update content of R and B*
$R := s(V^{\mathsf{R}})$; $B := \emptyset$
for $v \in V^{\mathsf{B}}$ **do**
  $B := B\cup$values reachable in v's DTG from $I(v)$ using only outside conditions from R
endfor

Procedure: Achieve(s, g) *// Same as slide 19*

Reduced Over-Estimation

Initial state heuristic values:



[FF: h^{FF}; AFS: Relaxed Plan Repair; ARS: Relaxed Facts Following]

Improved Performance

Coverage (instances solved), for painting strategies "A" vs. "C":

	#	FF	AF	AR	CF	CR
barman	20/20	15	16	16	17	2
depot	22/22	15	14	15	14	15
driverlog	20/20	18	16	18	17	18
elevators	20/20	17	14	13	2	11
floortile	20/20	4	6	3	6	3
grid	5/5	4	3	4	4	4
logistics98	35/35	22	5	35	5	35
mprime	35/35	30	31	30	29	30
nomystery	20/20	8	7	14	7	14
parcprinter	13/20	4	6	4	6	4
Pipes-notank	40/50	20	18	18	18	18
Pipes-tank	40/50	14	16	12	16	13
rovers	40/40	23	16	25	17	25
satellite	36/36	23	22	28	22	28
scanalyzer	14/20	10	12	14	10	10
sokoban	20/20	19	19	19	18	19
tidybot	20/20	15	14	13	16	13
tpp	30/30	20	15	20	15	20
transport	20/20	0	0	0	1	0
trucks	30/30	16	15	16	16	14
visitall	20/20	5	3	17	3	17
woodworking	20/20	2	2	3	2	3
$\sum$	891/926	644	610	677	601	656

[AF, CF: Relaxed Plan Repair; AR, CR: Relaxed Facts Following]

Questionnaire

Consider the same relaxed plan: $drive(g,a)$, $drive(g,b)$, $drive(g,c)$, $drive(g,d)$, $load(A,a)$, $load(B,b)$, $load(C,c)$, $load(D,d)$, $unload(A,g)$, $unload(B,g)$, $unload(C,g)$, $unload(D,g)$.

Question!

What does Relaxed Facts Following do if, in Star-Shape Logistics, instead of the truck we paint the packages black?

(A): Nothing

(B): Same as Before

$\rightarrow$ The R^+ facts – red preconditions/goals achieved by $\vec{a}^+$ – are now $\{v_T = a, v_T = b, v_T = c, v_T = d\}$. To achieve these, the only actions A' that can be used are $drive(g,x)$ so these are selected, and directly executed because they don't have any black preconditions. Having thus achieved R^+ by driving the truck across the map, the algorithm proceeds to repair the black goals, namely $\{v_a = g, v_b = g, v_c = g, v_d = g\}$. For each of these, the Achieve procedure selects the DTG path induced by $load(X,x), unload(X,g)$.

$\rightarrow$ So Relaxed Facts Following re-produces exactly the relaxed plan we started out with. This is not what one would expect as the meaning of "Nothing" or "Same as Before", but both statements could be interpreted with that meaning.

Before We Begin

Which Other Methods? Apart from red-black relaxation, there are two other methods that allow to smoothly interpolate between h^+ and h^*:

- **Variable Pre-Merging:** Use $h^+_{\Pi^M}$ where Π^M is obtained from Π by merging a subset M of variables into a single variable.
- **Conjuncts Compilation:** Use $h^+_{\Pi^C}$ where Π^C is obtained from Π by explicitly representing a subset C of fact conjunctions.

Illustrative example we will use here: Buy-A-Car

vs.

- State variables: C, G : Boolean.
- Initial state: $C = 0, G = 1$.
- Goal: $C = 1, G = 1$.
- Action: $buy()$
  Precondition $C = 0, G = 1$; effect $C = 1, G = 0$.

So what? Task is unsolvable but has relaxed plan $buy()$.

Variable Pre-Merging

Method outline:

- Before planning starts, select a subset $M \subseteq V$ of FDR variables.
- Compute the DTG of a merged variable x_M equivalent to the cross-product of M.
- Replace M with x_M in the planning task Π to obtain the merged task Π^M.

Applied to Buy-A-Car:

- $M := \{C, G\}$; $D_{x_M} = \{C0G0, C0G1, C1G0, C1G1\}$.
- $I(x_M) = C0G1$; $G(x_M) = C1G1$.
- Value transitions on x_M: Only $C0G1 \xrightarrow{buy()} C1G0$.
- Relaxed plan for Π^M: None: No path from $I(x_M)$ to $G(x_M)$.
  $\rightarrow$ So we have $h^+_{\Pi} = 1 \ll \infty = h^+_{\Pi^M} = h^*_{\Pi}$.

Variable Pre-Merging: Convergence

Proposition (Variable Pre-Merging is Perfect in the Limit). *Let $\Pi = (V, A, c, I, G)$ be an FDR planning task. If we set $M := V$ in the above, then $h^+_{\Pi^M} = h^*_\Pi$.*

Proof. If we merge all variables, then the merged task has a single variable whose DTG is the overall state space. A relaxed plan through that DTG is a solution path in the state space, QED.

$\rightarrow$ Problem with that result? The "Limit" case is trivial and involves building the whole state space in the first place.

Explicit Conjunctions: Idea

Method outline:

- Before planning on FDR task $\Pi = (V, A, c, I, G)$ starts, select a subset C of fact conjunctions c to be represented explicitly using new π-fluents π_c.

  $\rightarrow$ E.g., $C = \{p \wedge q, g_1 \wedge g_2\}$ and we introduce new Boolean variables $\pi_{p\wedge q}$ and $\pi_{g_1 \wedge g_2}$.
- Construct a compiled task Π^C, modifying Π to correctly account for the intended semantics of each π_c.
- Initial state: Include those π_c where $c \subseteq I$. (We identify conjunctions with sets of facts.)
- Action effects: If eff_a intersects c and does not contradict c, then make a copy of a whose effect includes π_c.
- Action preconditions and goal: In Π^C, include each π_c into every condition (precondition/goal) that contains c.

Explicit Conjunctions in Buy-A-Car

VS.

- State variables: v_C, v_G : Boolean.
- Initial state: $v_C = 0, v_G = 1$.
- Goal: $v_C = 1, v_G = 1$.
- Actions (unit costs): $buy()$
  Precondition $v_C = 0, v_G = 1$; effect $v_C = 1, v_G = 0$.

Now let's make one conjunction explicit:

- $C :=$ set of conjunctions containing only $c := v_C = 1 \wedge v_G = 1$.
- Goal of Π^C: $\{v_C = 1, v_G = 1, \pi_c\}$.
- Actions a where eff_a intersects c and does not contradict c: None. $buy()$ achieves $v_C = 1$ but contradicts $v_G = 1$.
- Relaxed plan for Π^C: None. No action achieves the goal fact π_c.

The Π^C Compilation

Shorthand notation: For fact set X, $X^C := X \cup \{\pi_c \mid c \in C, c \subseteq X\}$.

Definition (The Π^C compilation). *Let $\Pi = (V, A, c, I, G)$ be an FDR planning task, and let C be a set of conjunctions (fact sets/partial variable assignments) in Π. Then Π^C is the task $(V^C, A^C, c^C, I^C, G^C)$ where:*

- *$V^C = V \cup \{\pi_c \mid c \in C\}$, each π_c being a new Boolean variable.*
- *A^C contains an action $a^{C'}$ for every pair $a \in A$, $C' \subseteq C$ s.t., for all $c' \in C'$, $eff_a \cap c' \neq \emptyset$ and there is no $v \in V$ s.t. $eff_a(v)$ and $c(v)$ are both defined and $eff_a(v) \neq c(v)$; $a^{C'}$ is then given by*
  - *$pre_{a^{C'}} = [pre_a \cup \bigcup_{c' \in C'}(c' \setminus eff_a)]^C$, and*
  - *$eff_{a^{C'}} = eff_a \cup \{\pi_{c'} \mid c' \in C'\}$.*
- *c^C extends c to A^C by $c(a^{C'}) = c(a)$.*
- *I^C and G^C are as defined by the shorthand notation.*

$\rightarrow$ Action a can achieve conjunctions C', at the cost of having the "missing context" $\bigcup_{c' \in C'}(c' \setminus eff_a)$ beforehand.

The Π^C Compilation: Why "every pair $a \in A$, $C' \subseteq C$"?

What is the growth of Π^C in $|C|$? Exponential! We enumerate subsets $C' \subseteq C$.

Why do we need this? Why don't we only include a^c for $a \in A$, $c \in C$ that a can support? Because this would lose admissibility.

Example where $h^+_{\Pi^C}$ would be $> h^*_\Pi$: (Notation here STRIPS-like; read as "FDR with Boolean variables" if you prefer)

Facts: $\{q_1, q_2, p, g_1, g_2\}$; initial state: $\emptyset$; goal: $\{g_1, g_2\}$. Actions:

- $a_{q_1} : \emptyset \to q_1, \neg p \quad a_{q_2} : \emptyset \to q_2, \neg p$
- $a_p : \emptyset \to p$
- $a_{g_1} : p, q_1 \to g_1 \quad a_{g_2} : p, q_2 \to g_2$

$\to$ Say we use $C := \{p \wedge q_1, p \wedge q_2\}$. Then a_{g_1} has precondition $\{p, q_1, \pi_{p \wedge q_1}\}$ and a_{g_2} has precondition $\{p, q_2, \pi_{p \wedge q_2}\}$. Say Π^C includes the actions $a_p^{\{p \wedge q_1\}}$ and $a_p^{\{p \wedge q_2\}}$, but does not include $a_p^{\{p \wedge q_1, p \wedge q_2\}}$. Then $h^+_{\Pi^C}(I) = 6 > 5 = h^*(I)$.

The Π^C Compilation: Example for $h^+ < h^+_{\Pi^C} < \infty$

Example from previous slide, modified to have conflict between q_1 and q_2:

Facts: $\{q_1, q_2, p, g_1, g_2\}$; initial state: $\emptyset$; goal: $\{g_1, g_2\}$. Actions:

- $a_{q_1} : \emptyset \to q_1, \neg p, \neg q_2 \quad a_{q_2} : \emptyset \to q_2, \neg p, \neg q_1$
- $a_p : \emptyset \to p$
- $a_{g_1} : p, q_1 \to g_1 \quad a_{g_2} : p, q_2 \to g_2$

$\to$ Plan? $a_{q_1}, a_p, a_{g_1}, a_{q_2}, a_p, a_{g_2}$.

$\to$ Relaxed plan? $a_{q_1}, a_{q_2}, a_p, a_{g_1}, a_{g_2}$.

$\to$ Relaxed plan for Π^C when taking $C := \{p \wedge q_1, p \wedge q_2, q_1 \wedge q_2\}$?

- Can we do $a_{q_1}, a_{q_2}, a_p^{\{p \wedge q_1, p \wedge q_2\}}, a_{g_1}, a_{g_2}$? No, because $a_p^{\{p \wedge q_1, p \wedge q_2\}}$ has the precondition π_{q_1, q_2}, which is unreachable.
- So how to do it? $a_{q_1}, a_p^{\{p \wedge q_1\}}, a_{g_1}, a_{q_2}, a_p^{\{p \wedge q_2\}}, a_{g_2}$, like real plan above.

Explicit Conjunctions: Convergence

Theorem (The Π^C Compilation is Perfect in the Limit). *Let Π be an FDR planning task. Then there exists C such that $h^+_{\Pi^C} = h^*_\Pi$.*

Proof. For sufficiently large m, $h^m_\Pi = h^*_\Pi$ **(Chapter 8)**. If we choose C to be all size-$\leq m$ conjunctions, then $h^m_\Pi = h^1_{\Pi^C}$ [see e.g. Keyder *et al.* (2012)]. Done with $h^1_{\Pi^C} = h^{\max}_{\Pi^C} \leq h^+_{\Pi^C}$ **(Chapter 9)** $\leq h^*_\Pi$.

Problem with that result: The "Limit" case, as proved here, is $h^m = h^*$ which typically happens only for prohibitively large m.

$\to$ However, the proof argument ignores the advantages of $h^+(\Pi^C)$:
1. We can choose C more freely.
2. we use h^+ instead of h^1.

So there is hope to obtain h^* with much smaller C. (See slide 47)

So Which Method Should We Use?

Short answer: Nobody knows.

Longer answer:

- Implemented methods so far have varying strengths and weaknesses, there is no clear winner, except variable pre-merging performs worse (so far) than red-black planning and explicit conjunctions.
- Theory comparison: Which methods can/cannot *be simulated* by which other ones?

  [Hoffmann *et al.* (2014)]: None of $h^+(\Pi^C)$, red-black planning, and variable pre-merging can simulate any other with polynomial overhead, except that $h^+(\Pi^C)$ simulates pre-merging variables M when setting C to contain all fact conjunctions c over M.

Summary

- The delete relaxation is unable to account for to-and-fro, and for harmful side effects. To counter-act this, we should "take some deletes into account". If such a method is able to render h^+ perfect in the limit, then we call it an interpolation method.
- Red-black planning is an interpolation method that relaxes only a subset of the FDR state variables (the red variables), keeping the others (the black variables) intact.
- Red-black planning is **NP**-hard even with a single black variable, but is tractable if we demand ("SMS Theorem") that the black causal graph is acyclic, and that all black variables are invertible.
- Naïve red-black planning by Relaxed Plan Repair is prone to over-estimation, but we can fix this by relying less on the relaxed plan in Relaxed Facts Following.
- Explicit conjunctions is an alternative interpolation method, expliciting the semantics of a subset C of conjunctions over the task's facts.

Remarks

Beyond the SMS theorem: I've treated you to this simple setup for simplicity.

- Our actual theorem is more general in requiring only an acyclic black causal graph, instead of requiring there to be no arcs at all.
- Our actual theorem is more general in requiring only "relaxed side-effects invertibility", a weaker notion of invertibility.
- There's an alternative tractability theorem, requiring only that the domain size of the (single) black variable is bounded.

Painting strategies: Which variables to paint red respectively black?

- We experimented with lots of methods based on different notions of which variables are "most important" (to be painted black as much as possible).
- The performance differences are, generally speaking, marginal.
- In fact, there typically is very little choice if we insist on painting black "as much as possible".
- Comprehensive results: [Domshlak *et al.* (2015)]

Research → FAI BSc/MSc/HiWi

... (a few examples) ...

Theory Understanding:

- Identify special cases where polynomial-size C can/cannot render $h^+_{\Pi^C}$ perfect.
- Deeper complexity analysis of red-black planning.
- Generalizations of red-black planning where variables may remember *some* of their values.
- Etc. ...

Alternative Uses of Partial Delete Relaxation:

- Learning to detect dead-ends [Steinmetz and Hoffmann (2016)]/learning to refine heuristic values during search.
- Incremental red-black.
- Plan templates to seed plan-space search.
- Plan-template distance heuristics.

Reading

- *Who Said we Need to Relax All Variables?* [Katz *et al.* (2013b)].

  Available at:

  `http://fai.cs.uni-saarland.de/hoffmann/papers/icaps13.pdf`

  Content: Introduces red-black planning and our main complexity results, along with a brief analysis of when/where $h^{*\mathrm{RB}}$ is perfect.

- *Red-Black Relaxed Plan Heuristics* [Katz *et al.* (2013a)].

  Available at:

  `http://fai.cs.uni-saarland.de/hoffmann/papers/aaai13.pdf`

  Content: Simpler tractable fragment (SMS Theorem + relaxed side-effects invertibility) used to generate red-black plan heuristics.

Reading

- *Red-Black Relaxed Plan Heuristics Reloaded* [Katz and Hoffmann (2013)].

  Available at:
  http://fai.cs.uni-saarland.de/hoffmann/papers/socs13.pdf

  Content: As above, but with Relaxed Facts Following for reduced over-estimation and (much) better performance.

- *Red-Black Planning: A New Systematic Approach to Partial Delete Relaxation* [Domshlak *et al.* (2015)].

  Available at:
  http://fai.cs.uni-saarland.de/hoffmann/papers/ai15.pdf

  Content: The whole storyline of the previous three papers, comprehensively told and underfed with systematic experiments.

Reading, ctd.

- *Improving Delete Relaxation Heuristics Through Explicitly Represented Conjunctions* [Keyder *et al.* (2014)].

  Available at:
  http://fai.cs.uni-saarland.de/hoffmann/papers/jair14.pdf

  Content: Uses the Π^C compilation as well as another compilation Π^C_{ce} which employs conditional effects to avoid the exponential blow-up in $|C|$. This comes at the prize of a loss in informedness, however Π^C_{ce} is still perfect in the limit.

- *Combining the Delete Relaxation with Critical-Path Heuristics: A Direct Characterization* [Fickert *et al.* (2016)].

  Available at:
  http://fai.cs.uni-saarland.de/hoffmann/papers/jair16.pdf

  Content: Avoids the compilation altogether. Achieves the same complexity reduction as Π^C_{ce}, but without the information loss.

References I

Carmel Domshlak, Jörg Hoffmann, and Michael Katz. Red-black planning: A new systematic approach to partial delete relaxation. *Artificial Intelligence*, 221:73–114, 2015.

Maximilian Fickert, Jörg Hoffmann, and Marcel Steinmetz. Combining the delete relaxation with critical-path heuristics: A direct characterization. *Journal of Artificial Intelligence Research*, 56(1):269–327, 2016.

Jörg Hoffmann, Marcel Steinmetz, and Patrik Haslum. What does it take to render $h^+(\pi^c)$ perfect? In *ICAPS 2014 Workshop on Heuristics and Search for Domain-Independent Planning (HSDIP'14)*, 2014.

Michael Katz and Jörg Hoffmann. Red-black relaxed plan heuristics reloaded. In Malte Helmert and Gabriele Röger, editors, *Proceedings of the 6th Annual Symposium on Combinatorial Search (SOCS'13)*, pages 105–113. AAAI Press, 2013.

Michael Katz, Jörg Hoffmann, and Carmel Domshlak. Red-black relaxed plan heuristics. In Marie desJardins and Michael Littman, editors, *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI'13)*, pages 489–495, Bellevue, WA, USA, July 2013. AAAI Press.

References II

Michael Katz, Jörg Hoffmann, and Carmel Domshlak. Who said we need to relax *all* variables? In Daniel Borrajo, Simone Fratini, Subbarao Kambhampati, and Angelo Oddi, editors, *Proceedings of the 23rd International Conference on Automated Planning and Scheduling (ICAPS'13)*, pages 126–134, Rome, Italy, 2013. AAAI Press.

Emil Keyder, Jörg Hoffmann, and Patrik Haslum. Semi-relaxed plan heuristics. In Blai Bonet, Lee McCluskey, José Reinaldo Silva, and Brian Williams, editors, *Proceedings of the 22nd International Conference on Automated Planning and Scheduling (ICAPS'12)*, pages 128–136. AAAI Press, 2012.

Emil Keyder, Jörg Hoffmann, and Patrik Haslum. Improving delete relaxation heuristics through explicitly represented conjunctions. *Journal of Artificial Intelligence Research*, 50:487–533, 2014.

Marcel Steinmetz and Jörg Hoffmann. Towards clause-learning state space search: Learning to recognize dead-ends. In Dale Schuurmans and Michael Wellman, editors, *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, February 2016.