Optimising Partial-Order Plans Via Action Reinstantiation

Max Waters , **Lin Padgham** and **Sebastian Sardina**
RMIT University, Melbourne, Victoria 3000, Australia
{max.waters, lin.padgham, sebastian.sardina}@rmit.edu.au

Abstract

This work investigates the problem of optimising a partial-order plan's (POP) flexibility through the simultaneous transformation of its action ordering and variable binding constraints. While the former has been extensively studied through the notions of *deordering* and *reordering*, the latter has received much less attention. We show that a plan's variable bindings are often related to resource usage and their reinstantiation can yield more flexible plans. To do so, we extend existing POP optimality criteria to support variable reinstantiation, and prove that checking if a plan can be optimised further is NP-complete. We also propose a MAXSAT-based technique for increasing plan flexibility and provide a thorough experimental evaluation that suggests that there are benefits in action reinstantiation.

1 Introduction

This work investigates the modification of objects and resources in a plan to achieve higher flexibility *w.r.t.* action orderings. While the optimisation of action ordering has been extensively studied through the notions of *plan deordering* and *plan reordering*, both from a theoretical [Bäckström, 1998; Aghighi and Bäckström, 2017] and practical [Kambhampati and Kedar, 1994; Muise *et al.*, 2016; Siddiqui and Haslum, 2012; Say *et al.*, 2016] perspective, the optimisation of resources (or actions' domain objects) that are used in the course of executing a plan has received much less attention.

Ultimately, the aim is to achieve a *least commitment* approach to planning [Weld, 1994] that maximises execution-time flexibility by delaying decisions regarding action ordering and resource utilisation for as long as possible. Consider a planning instance from (a reduced version of) the IPC *rovers* domain, in which a fleet of rovers must navigate the surface of a planet, collecting soil and rock samples. The domain objects comprise two rovers (R_1 and R_2) and three waypoints (W_1, W_2, and W_3). There are soil and rock samples at waypoints W_2 and W_3, resp., and both rovers begin at W_1. The goal is to gather both samples. Plan P_1 in Figure 1a is an optimal solution: R_1 navigates from W_1 to W_2 and collects the soil sample, then navigates to W_3 and collects the rock sample. An examination of the causal structure of P_1 shows that

1. $move_1(\mathsf{R}_1, \mathsf{W}_1, \mathsf{W}_2)$
2. $get\text{-}soil(\mathsf{R}_1, \mathsf{W}_2)$
3. $move_2(\mathsf{R}_1, \mathsf{W}_2, \mathsf{W}_3)$
4. $get\text{-}rock(\mathsf{R}_1, \mathsf{W}_3)$

(a) Plan P_1.

1. $move_1(\mathsf{R}_1, \mathsf{W}_1, \mathsf{W}_2)$
2. $get\text{-}soil(\mathsf{R}_1, \mathsf{W}_2)$
3. $move_2(\mathsf{R}_2, \mathsf{W}_1, \mathsf{W}_3)$
4. $get\text{-}rock(\mathsf{R}_2, \mathsf{W}_3)$

(b) Plan P_2.

$$\begin{aligned}
O = \{\, & move_1(x_1, x_2, x_3) && \theta = \{\, x_1\backslash\mathsf{R}_1, x_2\backslash\mathsf{W}_1, x_3\backslash\mathsf{W}_2\\
& get\text{-}soil(x_4, x_5) && \quad x_4\backslash\mathsf{R}_1, x_5\backslash\mathsf{W}_2\\
& move_2(x_6, x_7, x_8) && \quad x_6\backslash\mathsf{R}_2, x_7\backslash\mathsf{W}_1, x_8\backslash\mathsf{W}_3\\
& get\text{-}rock(x_9, x_{10})\,\} && \quad x_9\backslash\mathsf{R}_2, x_{10}\backslash\mathsf{W}_3\,\}\\
\prec = \{\, & move_1 \prec get\text{-}soil, move_2 \prec get\text{-}rock\,\}
\end{aligned}$$

(c) Partial-order plan P_3.

Figure 1: Three plans from the *rovers* domain. Subscripts have been used to better distinguish between actions of the same type.

the order of its actions cannot be altered without compromising its validity (e.g., step 3 *threatens* the causal link between steps 1 and 2, as it moves R_1 away from W_2).

Plan P_2 in Figure 1b is a modification of P_1 that instead uses rover R_2 to collect the rock sample. This modification allows the plan's ordering constraints to be relaxed: the actions can be executed in any order so long as $1 \prec 2$ and $3 \prec 4$.

This simple example shows that an over-commitment to a particular set of resources (in this case, R_1) can reduce a plan's possible orderings. Thus, this paper introduces the notions of *reinstantiated deorderings* and *reinstantiated reorderings*, namely, transformations of a plan under which ordering constraints can be removed or arbitrarily modified, resp., *and the plan's actions' variable bindings can be changed*. The idea is to seek alternative variable bindings supporting greater minimisation of the plan's ordering constraints than is possible if the bindings remained unchanged.

More concretely, we define optimality criteria for partial-order plans (POP) that take variable bindings into consideration (Section 3) and propose a technique for computing optimally reinstantiated POPs by encoding the problem into a MAXSAT instance (Section 4). Finally, we experimentally assess the benefits of reinstantiation by comparing it with existing de/reordering techniques (Section 5). Results show that optimising both bindings and orderings incurs a much greater computational cost than optimising orderings alone. However, when (even non-optimal) reinstantiated de/reorderings can be found, they are significantly more flexible than plans produced by standard optimisation methods.

2 Preliminaries

2.1 Logical Preliminaries

The logical structures will be expressed in a function-free first-order language $\mathcal{L}$ with finitely many variable, constant, and predicate symbols, and the usual logical connectives, including equality and precedence [Brachman and Levesque, 2004, Chapter 2]. The letters x, y and z indicate variables, c constants and t, u and v terms (all possibly with annotations). The notation $\vec{t}$ denotes ordered lists of possibly non-unique terms, with $\vec{t}[i]$ indicating the i-th element of the list and $\vec{t_1} = \vec{t_2}$ being shorthand for $|\,\vec{t_1}\,| = |\,\vec{t_2}\,| \wedge \vec{t_1}[1] = \vec{t_2}[1] \wedge \cdots \wedge \vec{t_1}[|\,\vec{t_1}\,|] = \vec{t_2}[|\,\vec{t_2}\,|]$. For any structure η expressed in $\mathcal{L}$, $\mathsf{vars}(\eta)$ and $\mathsf{consts}(\eta)$ denote the variables and constants appearing in η, resp.

A *substitution* θ is a mapping from variables to terms, for example $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$ maps each variable x_i to term t_i, for $1 \leq i \leq n$, and every other variable to itself. We use $\mathsf{domain}(\theta)$ to denote the set of variables explicitly mapped by substitution θ, and $\theta(x)$ to denote the term corresponding to variable x under θ. The result of applying a substitution θ to a logical structure η is written $\eta\theta$, and means to simultaneously replace every variable x in η with $\theta(x)$. A substitution θ is *ground* if every variable in its domain is mapped to a ground term, and is *complete w.r.t.* η *iff* $\mathsf{vars}(\eta) \subseteq \mathsf{domain}(\theta)$.

2.2 Classical Planning Formalism

Classical planning tasks will be expressed in a standard first-order STRIPS formalism [Lifschitz, 1987].

Classical Planning Tasks

A *classical planning task* is a tuple $\Pi = \langle \mathcal{C}, \mathcal{O}, s_I, s_G \rangle$, where $\mathcal{C}$ is a set of constants, $\mathcal{O}$ is a set of operators, and s_I and s_G are sets of ground literals describing the initial state and goal, respectively. An *operator* is a tuple $o = \langle \mathsf{name}(o), \mathsf{vars}(o), \mathsf{pre}(o), \mathsf{post}(o) \rangle$, where $\mathsf{name}(o)$ is the name, or *type* of the operator, $\mathsf{vars}(o)$ is a list of variables and $\mathsf{pre}(o)$ and $\mathsf{post}(o)$ are finite sets of (ground or non-ground) literals with variables taken from $\mathsf{vars}(o)$. It will be assumed that for all operators, $\mathsf{name}(o) = o$, and so operators will be referred to by name. When distinguishing between different operators of the same name, the notation $o(\vec{x})$ is used, where $\mathsf{vars}(o) = \vec{x}$. An *action* is a ground operator $a = \langle \mathsf{pre}(a), \mathsf{post}(a) \rangle$, where $\mathsf{pre}(a)$ and $\mathsf{post}(a)$ are finite sets of ground literals. If o is an operator and substitution θ is ground and complete *w.r.t.* $\mathsf{vars}(o)$, then $\langle \mathsf{pre}(o)\theta, \mathsf{post}(o)\theta \rangle$ is the action resulting from *instantiating* o with θ.

Classical and Partial-Order Plans

A *classical plan* $\vec{a}$ is a finite sequence of actions. Assuming that no variable appears in more than one operator, a plan can also be represented as $\vec{o}\theta$, where $\vec{o}$ is a list of operators and θ is a ground substitution that is complete *w.r.t.* $\vec{o}$. A planning task can be "embedded" into a plan by including the distinguished actions a_I and a_G whose pre/postconditions are the task's initial state and goal, resp. In this form, a classical plan is valid *iff* it is *executable* (i.e., the actions can be applied in sequence without violating their precondition requirements).

A *partial-order plan* (POP) is a generalised classical plan that need not completely define the order of its operators:

Definition 1. *A* **partial-order plan** *(POP) is a tuple* $P = \langle O, \theta, \prec \rangle$ *where* O *is a set of operators,* θ *is a ground substitution which is complete* w.r.t. *to* O*, and* $\prec$ *is a strict, transitively closed partial order over* O.

As with classical plans, a planning task is "embedded" into a POP via the inclusion of the distinguished parameter-free operators o_I (which yields the initial state) and o_G (which checks the goals). A POP is a compact representation of a set of classical plans, known as its *linearisations*:

Definition 2. *A classical plan* $\langle o_1, \ldots, o_n \rangle\theta$ *is a* **linearisation** *of POP* $P = \langle O, \theta, \prec \rangle$ iff $O = \{o_1, \ldots, o_n\}$ *and* $\prec \subseteq \{a_i \prec a_j : 1 \leq i < j \leq n\}$.

A POP's *validity* is defined *w.r.t.* its linearisations:

Definition 3. *A POP is* **valid** iff *all of its linearisations are executable.*

Classical plans are special cases of POPs, and any classical plan $\langle o_1, \ldots, o_n \rangle\theta$ can be expressed as an equivalent partial-order plan $\langle \{o_1, \ldots, o_n\}, \theta, \{a_i \prec a_j : 1 \leq i < j \leq n\} \rangle$.

The Producer-Consumer-Threat Formalism

The *producer-consumer-threat formalism* (PCT) [Bäckström, 1998] is typically used to describe a POP's causal structure by identifying which actions produce, consume or threaten which ground literals. Here, it describes how operators produce, consume or threaten (possibly non-ground) literals:

Definition 4. *Let* o *be an operator and* $q(\vec{t})$ *a literal. Then:*

$$\mathsf{prods}(o, q(\vec{t})) \stackrel{\text{def}}{=} q(\vec{t}) \in \mathsf{post}(o).$$

$$\mathsf{cons}(o, q(\vec{t})) \stackrel{\text{def}}{=} q(\vec{t}) \in \mathsf{pre}(o).$$

$$\mathsf{thrtns}(o, q(\vec{t})) \stackrel{\text{def}}{=} \neg q(\vec{t}) \in \mathsf{post}(o).$$

In the field of *partial-order causal link planning* (POCL), a PCT-based notion of POP validity is used that derives from the implied causal dependencies between a POP's operators [Weld, 1994], not the validity of its linearisations. A *causal link* associates a consumer, (i.e., an operator precondition), with a producer (i.e., an operator postcondition):

Definition 5. *A* **causal link** *is a* 4*-tuple* $\langle o_p, q(\vec{t}), o_c, q(\vec{u}) \rangle$ s.t. $\mathsf{prods}(o_p, q(\vec{t}))$ *and* $\mathsf{cons}(o_c, q(\vec{u}))$.

The consumer is *supported* by the causal link *iff* it is preceded by, and codesignated with, the producer (i.e., $\theta(\vec{t}) = \theta(\vec{u})$). The causal link is *threatened iff* a codesignated threat (i.e., some $o_t, q(\vec{v})$ *s.t.* $\mathsf{thrtns}(o_t, q(\vec{v}))$ and $\theta(\vec{v}) = \theta(\vec{u})$) can be ordered between the producer and consumer. A POP P's implicit *unthreatened causal links* are denoted L_P, and defined as follows:

Definition 6. *The* **unthreatened causal links** *of a POP* $P = \langle O, \theta, \prec \rangle$ *is the set* L_P s.t. $\langle o_p, q(\vec{t}), o_c, q(\vec{u}) \rangle \in L_P$ iff*:*

1. $o_p \prec o_c$,
2. $\theta(\vec{t}) = \theta(\vec{u})$*, and*
3. *for all* o_t, $q(\vec{v})$ s.t. $\mathsf{thrtns}(o_t, q(\vec{v}))$*, either* $\theta(\vec{u}) \neq \theta(\vec{v})$*,* $o_t \prec o_p$ *or* $o_c \prec o_t$.

A POP P is *POCL-valid iff* every consumer is supported by an unthreatened causal link:

Definition 7. *A POP $P = \langle O, \theta, \prec \rangle$ is* **POCL-valid** iff *for all $o_c \in O$ and $q(\vec{u})$* s.t. $\mathsf{cons}(o_c, q(\vec{u}))$, *there exist $o_p \in O$ and $q(\vec{t})$* s.t. $\langle o_p, q(\vec{t}), o_c, q(\vec{u}) \rangle \in L_P$.

This notion of validity is stronger than that in Definition 3, meaning that a POP might not be POCL-valid despite its linearisations being executable [Kambhampati and Nau, 1996]. However, any such POP can always be made POCL-valid by adding ordering constraints [Bercher and Olz, 2020].

2.3 Partial Weighted MaxSAT

The *partial weighted maximum satisfiability* (partial weighted MaxSAT) problem is a generalised optimisation variant of SAT that distinguishes between *soft* clauses, which are assigned a numeric weight, and *hard* clauses, which are unweighted. The aim of the partial weighted MaxSAT problem is to find an interpretation that satisfies all hard clauses and maximises the total weight of the satisfied soft clauses. The syntax $\overset{k}{\phi}$ indicates that clause ϕ has weight k, and no weight marking indicates a hard clause. For example, the formula $\overset{3}{(p_1 \vee p_2)} \wedge \overset{2}{\neg p_1} \wedge \neg p_2$ is trivially unsatisfiable, but the interpretation $\{p_1/\top, p_2/\bot\}$ is an optimal solution to the partial weighted MaxSAT problem.

3 Optimality Criteria for Partial-Order Plans

Bäckström [1998]'s seminal work on action ordering and plan flexibility considers two types of modification: *deordering*, in which constraints can be removed but not added, and *reordering*, which allows both. Both must result in a *valid* POP:

Definition 8. *Let $P = \langle O, \theta, \prec \rangle$ and $Q = \langle O, \theta, \prec' \rangle$ be two POPs. Then:*

- *Q is a* **reordering** *of P* iff *they are both valid.*
- *Q is a* **deordering** *of P* iff *it is a reordering and $\prec' \subseteq \prec$.*
- *Q is a* **strict deordering** *of P* iff *it is a deordering and $\prec' \subset \prec$.*

Bäckström defines the relative optimality of two POPs by comparing their ordering constraints. A *minimal deordering* of a POP cannot be relaxed any further without rendering the POP invalid, and a *minimum deordering* is the smallest of all valid deorderings of a POP. A *minimum reordering* has the fewest possible ordering constraints while remaining valid:

Definition 9. *Let $P = \langle O, \theta, \prec \rangle$ and $Q = \langle O, \theta, \prec' \rangle$ be two POPs. Then:*

- *Q is a* **minimal deordering** *of P* iff *Q is a deordering of P and there is no POP R such that R is a strict deordering of Q.*
- *Q is a* **minimum deordering** *of P* iff *Q is a deordering of P and there is no POP $R = \langle O, \theta, \prec'' \rangle$ such that R is a deordering of P and $|\prec''| < |\prec'|$.*
- *Q is a* **minimum reordering** *of P* iff *Q is a reordering of P and there is no POP $R = \langle O, \theta, \prec'' \rangle$ such that R is a reordering of P and $|\prec''| < |\prec'|$.*

While a minimal deordering of a given POP can be found in polynomial time, deciding whether there exists a de/reordering with fewer than k ordering constraints is NP-complete, and finding a minimum de/reordering is NP-hard and cannot be approximated within a constant factor.

A key limitation of the optimality criteria above is the assumption that the POP's variable bindings remain *static*. We thus define more generalised criteria that *accommodate modifications to a POP's variable bindings*. In all cases, the resulting POP must remain both ground (i.e., its variable bindings must be complete and ground *w.r.t.* O) and valid:

Definition 10. *Let $P = \langle O, \theta, \prec \rangle$ and $Q = \langle O, \theta', \prec' \rangle$ be two POPs. Then:*

- *Q is a* **reinstantiated reordering** *of P* iff *both P and Q are valid.*
- *Q is a* **reinstantiated deordering** *of P* iff *Q is a reinstantiated reordering of P and $\prec' \subseteq \prec$.*
- *Q is a* **reinstantiated strict deordering** *of P* iff *Q is a reinstantiated deordering of P and $\prec' \subset \prec$.*

For example, plans P_1–P_3 in Figure 1 use the same operators, and so they are reinstantiated reorderings of each other, whereas P_3 is a reinstantiated strict deordering of P_1 and P_2.

The notion of a "least-constrained" POP can now be generalised to allow for changes in variable bindings. A *minimal reinstantiated deordering* of a POP is a reinstantiated deordering that cannot be relaxed any further: no modification to the POP's variable bindings will allow for the removal of any ordering constraints. A *minimum reinstantiated deordering* of a POP has the fewest ordering constraints of all of its reinstantiated deorderings. Of all the possible modifications to a POP's ordering and binding, a *minimum reinstantiated reordering* contains the fewest ordering constraints:

Definition 11. *Let $P = \langle O, \theta, \prec \rangle$ and $Q = \langle O, \theta', \prec' \rangle$ be two POPs. Then:*

- *Q is a* **minimal reinstantiated deordering** *of P* iff *Q is a reinstantiated deordering of P and there is no POP R such that R is a reinstantiated strict deordering of Q.*
- *Q is a* **minimum reinstantiated deordering** *of P* iff *Q is a reinstantiated deordering of P and there is no POP $R = \langle O, \theta'', \prec'' \rangle$ such that R is a reinstantiated deordering of P and $|\prec''| < |\prec'|$.*
- *Q is a* **minimum reinstantiated reordering** *of P* iff *Q is a reinstantiated reordering of P and there is no POP $R = \langle O, \theta'', \prec'' \rangle$ such that R is a reinstantiated reordering of P and $|\prec''| < |\prec'|$.*

For example, plan P_3 has two ordering constraints, and as there is no reinstantiation of P_1 which allows for fewer than this, P_3 is a minimum reinstantiated reordering of P_1.

Complexity Results

Deciding whether a POP has a reinstantiated de/reordering with fewer than k ordering constraints is NP-complete, and the optimisation problem of finding a minimum reinstantiated de/reordering cannot be approximated within a constant factor (proofs for minimum reinstantiated deordering are provided, those for reordering are a trivial modification):

Theorem 1. *Given a POP $P = \langle O, \theta, \prec\rangle$ and an integer $k > 0$, determining whether there exists a POP $Q = \langle O, \theta', \prec'\rangle$* s.t. *$Q$ is a reinstantiated deordering (or reordering) of P and $|\prec'| < k$ is* NP*-complete.*

Proof. For membership, guess a POP $Q = \langle O, \theta', \prec'\rangle$ and verify in P-time that Q is valid, $\prec' \subseteq \prec$ and $|\prec'| < k$. Hardness is by reduction from the NP-complete [Bäckström, 1998] decision problem of minimum deordering, which asks whether P has a deorder with $< k$ ordering constraints. Construct $P' = \langle O', \theta, \prec\rangle$ where $o \in O$ *iff* there exists a $o' \in O'$ *s.t.* $\mathsf{vars}(o') = \mathsf{vars}(o)$, $\mathsf{post}(o') = \mathsf{post}(o)$ and $\mathsf{pre}(o') = \mathsf{pre}(o) \cup \{x = \theta(x) : x \in \mathsf{vars}(o')\}$. As the operators in O' have preconditions that "fix" the variable bindings, P has a deorder with $< k$ ordering constraints *iff* P' has a reinstantiated deorder with $< k$ ordering constraints. □

Theorem 2. *The problem of finding a minimum reinstantiated deordering (or reordering) of a POP is not in* APX *unless* NP $\subseteq$ DTIME$(n^{poly\ log\ n})$.

Proof. Proof is by *reductio*. Assume that minimum reinstantiated deordering is in APX. Then there must be a function, A, that approximates it within a constant factor. As the reduction in the Theorem 1 proof preserves solutions, A also approximates minimum deordering within a constant factor, which is impossible unless NP $\subseteq$ DTIME$(n^{\text{poly log n}})$ [Bäckström, 1998]. □

Optimality Under POCL-validity

Under the additional requirement that the input and optimised POPs be POCL-valid, minimum de/reordering remain NP-complete (a trivial corollary of Theorem 4.8 in [Bäckström, 1998]). However, due to the stronger requirements of POCL-validity there exist minimum POCL-valid de/reorderings that can be further optimised while remaining valid under Definition 3 [Kambhampati and Nau, 1996]. These complexity and completeness results can be trivially extended to minimum reinstantiated POCL-valid de/reordering.

4 Partial Weighted MaxSAT Encoding

The problem of optimising a POP's ordering can be naturally expressed as an instance of the partial weighted MaxSAT problem (Section 2.3). This approach was introduced by Muise *et al.* [2016], whose MD and MR encodings transform a POP into MaxSAT instances with optimal solutions corresponding to minimum POCL-valid de/reorderings, resp.

This section introduces MRD and MRR: generalised encodings that *allow the POP to be reinstantiated*. Their optimal solutions therefore correspond to minimum POCL-valid reinstantiated de/reorderings, resp. They also optimise MD and MR by breaking symmetries and removing propositions and clauses that are not required to enforce plan validity.

We note due to the difficulty of expressing Definition 3 into MaxSAT, all four encodings instead preserve plan validity with the POCL requirements in Definition 7. While easier to encode, this raises the possiblity that the resulting optimised POPs contain unnecessary ordering constraints.

4.1 The MD and MR Encodings

MD and MR use two "types" of propositional variables. If $P = \langle O, \prec, \theta\rangle$ is the input POP, then for each pair of actions $a_1, a_2 \in O\theta$, a proposition $p_{a_1 \prec a_2}$ is introduced to indicate that a_1 must precede a_2 in the resulting POP. For all a_c, a_p and q such that a_c consumes q and a_p produces q, a proposition p_{a_p,q,a_c} is introduced to encode the requirement that in the final POP, a_p be causally linked to a_c with respect to q.

Formulae 1–3 ensure that the output POP is acyclic and transitively closed with all actions ordered between the initial state and goal:

$$\bigwedge_{a} \neg p_{a \prec a}. \tag{1}$$

$$\bigwedge_{a_1,a_2,a_3} p_{a_1 \prec a_2} \wedge p_{a_2 \prec a_3} \rightarrow p_{a_1 \prec a_3}. \tag{2}$$

$$\bigwedge_{a \notin \{a_I, a_G\}} p_{a_I \prec a} \wedge p_{a \prec a_G}. \tag{3}$$

Formulae 4 and 5 encode POCL-validity as in Definition 7:

$$\bigwedge_{\substack{a_p,a_c,q: \\ \mathsf{cons}(a_c,q), \\ \mathsf{prods}(a_p,q))}} \big(p_{a_p,q,a_c} \rightarrow \bigwedge_{\substack{a_t: a_t \neq a_c, \\ \mathsf{thrtns}(a_t,q)}} p_{a_t \prec a_p} \vee p_{a_c \prec a_t}\big). \tag{4}$$

$$\bigwedge_{\substack{a_c,q: \\ \mathsf{cons}(a_c,q)}} \bigvee_{a_p:\mathsf{prods}(a_p,q)} p_{a_p \prec a_c} \wedge p_{a_p,q,a_c}. \tag{5}$$

Formula 6 adds a soft unit clause with the *negation* of each ordering proposition, meaning that higher weight solutions will have fewer ordering constraints:

$$\bigwedge_{a_1,a_2}^{1} \neg p_{a_1 \prec a_2}. \tag{6}$$

Definition 12. MR *encodes a POP into a partial weighted* MaxSAT *instance through Formulae 1–6.*

The MD encoding extends MR with clauses that disallow any ordering constraints that were not in the input plan, thus forcing the resulting POP to be deordering of the input:

$$\bigwedge_{(a_1,a_2) \notin \prec} \neg p_{a_1 \prec a_2}. \tag{7}$$

Definition 13. MD *encodes a POP into a partial weighted* MaxSAT *instance through Formulae 1–7.*

4.2 The MRD and MRR Encodings

MRD and MRR allow the input POP to be reinstantiated, and so use different variable "types" to express the allowable rebindings. Three types are used, as per the following sets:

- $\mathcal{P}_\prec$ contains propositions of the form $p_{o_1 \prec o_2}$, which indicate that $o_1 \prec o_2$ in the resulting POP.
- $\mathcal{P}_\theta$ contains propositions of the form $p_{t=u}$, which encode a requirement that in the final POP $\theta(t) = \theta(u)$.

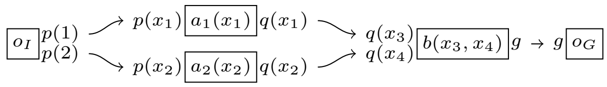

Figure 2: Plan P_4 from a synthetic domain. Pre/postconditions are to the left/right of operators, resp., and arrows indicate causal links.

- $\mathcal{P}_C$ contains propositions of the form $p_{o_p,\vec{t},o_c,\vec{u}}$, indicating that there must be some q *s.t.* causal link $\langle o_p, q(\vec{t}), o_c, q(\vec{u})\rangle$ is unthreatened in the final POP.

Muise *et al.* [2016] observed that the cubic number of clauses generated by Formula 2 make MD and MR infeasable for large plans. As this is exacerbated in MRD and MRR by the need to also close the variable equality relation, a number of optimisations are introduced.

The first removes any ordering or binding constraints (and propositions that encode them) that are not required to preserve causal links. For example, consider Plan P_4 in Figure 2. Operator a_1 cannot be used to produce any of a_2's preconditions, and nor does it threaten any of its postconditions (nor *vice-versa*). Therefore, no minimum de/reordering of P_4 will require that $a_1 \prec a_2$, $a_2 \prec a_1$ or that x_1 and x_2 be (non-) codesignated. MRD and MRR therefore exclude the propositions $p_{a_1 \prec a_2}$, $p_{a_2 \prec a_1}$, $p_{x_1 = x_2}$ and $p_{x_2 = x_1}$.

The second optimisation prevents the encoding of possible causal links if the required ordering constraints are known to be unsatisfiable. For example, Formula 7 in MD enforces deordering with negated unit clauses. Rather than include propositions that are *a priori* false, MRD removes the proposition and limits the possible causal links to those where the consumer did not precede the producer in the input POP. Additionally, POPs can contain multiple operators of the same type (e.g., the two instances of a in plan P_4), resulting in an exponential number of de/reorderings that are equivalent under permutation of the operators. As is standard [Say *et al.*, 2016; Shlyakhter, 2007], we (partially) break these symmetries by limiting ordering constraints between operators of the same type to those found in an arbitrary linearisation. However, rather than achieve this with additional symmetry breaking clauses, we limit $\mathcal{P}_\prec$ and $\mathcal{P}_C$ as above.

The definition below formalises and generalises these ideas. If o_1 and o_2 are operators and $\prec_A$ is a set of "allowable" orderings, then o_1 and o_2 are *causally related iff* $o_1 \prec_A o_2$ and there is a possible causal or threat relationship between them:

Definition 14. *o_1, o_2, $q(\vec{t})$ and $q(\vec{u})$ are* **causally related** w.r.t. $\prec_A$, iff $o_1 \prec_A o_2$ *and either:*

1. $\mathsf{prods}(o_1, q(\vec{t}))$ *and* $\mathsf{cons}(o_2, q(\vec{u}))$,
2. $\mathsf{thrtns}(o_1, q(\vec{t}))$ *and* $\mathsf{prods}(o_2, q(\vec{u}))$, *or*
3. $\mathsf{cons}(o_1, q(\vec{t}))$ *and* $\mathsf{thrtns}(o_2, q(\vec{u}))$.

The sets of propositional variables can now be defined:

Definition 15. *Let $P = \langle O, \theta, \prec\rangle$ be a POP and $\prec_A \subseteq O^2$ be a set of allowable orderings. Then $\mathcal{P}_\prec$, $\mathcal{P}_\theta$, and $\mathcal{P}_C$ are the smallest sets* s.t.*:*

- $p_{o_1 \prec o_2} \in \mathcal{P}_\prec$ iff *either* (i) *there exists a $q(\vec{t})$ and $q(\vec{u})$* s.t. *o_1, o_2, $q(\vec{t})$ and $q(\vec{u})$ are causally related* w.r.t. $\prec_A$, *or* (ii) *there exists an o_3* s.t. $p_{o_1 \prec o_3}, p_{o_3 \prec o_2} \in \mathcal{P}_\prec$.
- $p_{t=u} \in \mathcal{P}_\theta$ iff *either* (i) *there exists a o_1, o_2, $q(\vec{t})$ and $q(\vec{u})$ that are causally related* w.r.t. $\prec_A$ *and some i* s.t. $\vec{t}[i] = t$, $\vec{u}[i] = u$, *or* (ii) *there exists a v* s.t. $p_{t=v}, p_{v=u} \in \mathcal{P}_\theta$.[1]
- $p_{o_p,\vec{t},o_c,\vec{u}} \in \mathcal{P}_C$ iff *there exists a q* s.t. $\mathsf{prods}(o_p, q(\vec{t}))$, $\mathsf{cons}(o_c, q(\vec{u}))$ *and* $o_p \prec_A o_c$.

MRD and MRR encode a POP $P = \langle O, \theta, \prec\rangle$ and a set of allowable orderings $\prec_A \subseteq O^2$ through the following sets of clauses. Formulae 8–10 ensure that any solution represents an acyclic, transitively closed POP where all operators precede the goal and are preceded by the initial state. They derive from Formulae 1–3 in MR, but have removed any propositions and constraints not required to preserve validity:

$$\bigwedge_{p_{o_1 \prec o_2}, p_{o_2 \prec o_1} \in \mathcal{P}_\prec} \neg p_{o_1 \prec o_2} \lor \neg p_{o_2 \prec o_1}. \tag{8}$$

$$\bigwedge_{\substack{p_{o_1 \prec o_2}, p_{o_1 \prec o_3},\\ p_{o_2 \prec o_3} \in \mathcal{P}_\prec}} p_{o_1 \prec o_2} \land p_{o_2 \prec o_3} \rightarrow p_{o_1 \prec o_3}. \tag{9}$$

$$\bigwedge_{p_{o_I \prec o}, p_{o \prec o_G} \in \mathcal{P}_\prec} p_{o_I \prec o} \land p_{o \prec o_G}. \tag{10}$$

Formulae 11 and 12 ensures that the equality relation over variables and constants is symmetric and transitive, and Formula 13 ensures that each variable is bound to exactly one object. They are extensions to MD and MR that ensure the consistency of the POP's variable bindings:

$$\bigwedge_{p_{t=u} \in \mathcal{P}_\theta} p_{t=u} \leftrightarrow p_{u=t}. \tag{11}$$

$$\bigwedge_{\substack{p_{t=u}, p_{t=v},\\ p_{u=v} \in \mathcal{P}_\theta}} p_{t=u} \land p_{u=v} \rightarrow p_{t=v}. \tag{12}$$

$$\bigwedge_{x \in \mathsf{vars}(O)} \Big(\bigvee_{\substack{c \in \mathsf{consts}(O):\\ p_{x=c} \in \mathcal{P}_\theta}} p_{x=c} \quad \land \bigwedge_{\substack{c_1, c_2 \in \mathsf{consts}(O):\\ p_{x=c_1}, p_{x=c_2} \in \mathcal{P}_\theta,\\ c_1 \neq c_2}} \neg p_{x=c_1} \lor \neg p_{x=c_2} \Big). \tag{13}$$

Formulae 14 and 15 encode POCL-validity. Formula 14 requires that each consumer be causally linked to at least one producer, and Formula 15 defines the ordering, binding and threat protection constraints required for a causal link to hold. These derive from Formulae 4 and 5 in MD and MR, but have been optimised to remove redundant propositions, and generalised to allow the POP's variable bindings to change:

$$\bigwedge_{\substack{o_c, q(\vec{u}):\\ \mathsf{cons}(o_c, q(\vec{u}))}} \bigvee_{p_{o_p,\vec{t},o_c,\vec{u}} \in \mathcal{P}_C} p_{o_p,\vec{t},o_c,\vec{u}}. \tag{14}$$

$$\begin{aligned} &\bigwedge_{p_{o_p,\vec{t},o_c,\vec{u}} \in \mathcal{P}_C} p_{o_p,\vec{t},o_c,\vec{u}} \rightarrow \Big[\bigwedge_{1 \le i \le |\vec{t}|} p_{\vec{t}[i]=\vec{u}[i]} \land p_{o_p \prec o_c} \land \\ &\quad \bigwedge_{\substack{o_t, q(\vec{v}): o_t \neq o_c,\\ \mathsf{thrtns}(o_t, q(\vec{v}))}} \Big(\bigvee_{\substack{p \in \mathcal{P}_\prec \cap\\ \{p_{o_t \prec o_p}, p_{o_c \prec o_t}\}}} p \quad \lor \bigvee_{\substack{p \in \mathcal{P}_\theta \cap\\ \{p_{\vec{t}[i]=\vec{v}[i]} : 1 \le i \le |\vec{t}|\}}} \neg p \Big) \Big]. \end{aligned} \tag{15}$$

[1]Implemented code merges all $p_{t=u}$ and $p_{u=t}$ and updates Formulae 11 and 12 accordingly.

Formula 16, an optimised version of Formula 6 in MD and MR, adds soft clauses that minimise the ordering constraints:

$$\bigwedge_{p_{o_1 \prec o_2} \in \mathcal{P}_\prec}^{1} \neg p_{o_1 \prec o_2}. \tag{16}$$

The MRD and MRR encodings can now be defined. The allowed orderings for MRD are those from the input plan, forcing the resulting POP to be a deordering of the input:

Definition 16. MRD *encodes a POP* $P = \langle O, \theta, \prec \rangle$ *into a partial weighted* MAXSAT *instance through Formulae 8–16, with the allowable orderings* $\prec_A = \prec$.

The MRR encoding only disallows orderings for the purposes of symmetry breaking:

Definition 17. *If* $P = \langle O, \theta, \prec \rangle$ *is a POP and* $\prec'$ *is an arbitrary linearisation of* P*, then* MRR *encodes* P *into a partial-weighted* MAXSAT *instance through Formulae 8–16, with the allowable orderings* $\prec_A = O^2 \setminus \{(o_2, o_1) : \mathsf{name}(o_1) = \mathsf{name}(o_2), o_1 \prec' o_2\}$).

5 Experimental Evaluation

While a minimum reinstantiated de/reorder of a POP can never be less flexible than a minimum de/reorder (Definitions 9 and 11), the question remains whether, in practice, reinstantiation provides any significant increase in flexibility. The search for a minimum de/reorder often times out before an optimal (or indeed any) solution has been found [Muise *et al.*, 2016], and allowing variable bindings creates a harder problem with an exponentially larger search space. This evaluation will thus address the question of whether reinstantiated de/reordering can provide more flexibility than standard de/reordering under the same resource constraints.

5.1 Experimental Setup

We answer the question above by comparing optimised plans produced by the Loandra MAXSAT solver [Berg *et al.*, 2019] using the MD, MR, MRD and MRR encodings.

We also compare the MAXSAT-based techniques with the *explanation-based order generalisation* (EOG) deordering technique of Kambhampati and Kedar [1994]. First, a *validation structure* is constructed: a subset of L_P (Definition 6) that links each consumer to its earliest producer:

Definition 18. *If* $P = \langle O, \theta, \prec \rangle$ *is a POP then* V_P *is a* **validation stucture** *of* P iff $V_P \subseteq L_P$ *and there exists a total order* $\prec \subseteq \prec'$ s.t. *if* $\langle o_p, q(\vec{t}), o_c, q(\vec{u}) \rangle \in V_P$ *and* $\langle o'_p, q(\vec{t}'), o_c, q(\vec{u}) \rangle \in L_P$ *then* $o_p \prec' o'_p$.

This validation structure serves as a deordering heuristic:

Definition 19. *If* P *is a POP with validation structure* V_P*, then the* **explanation-based order generalisation** *of* P w.r.t. V_P *is the POP* $P' = \langle O, \theta, \prec'^{+} \rangle$ *where* $o_1 \prec' o_2$ iff *there exists a* $q(\vec{t})$, $q(\vec{u})$ s.t. *either:*

1. $\langle o_1, q(\vec{t}), o_2, q(\vec{u}) \rangle \in V_P$,
2. *there exists an* o_3 *and* $q(\vec{v})$ s.t. $\langle o_2, q(\vec{u}), o_3, q(\vec{v}) \rangle \in V_P$, $o_1 \prec o_2$ *and* $\mathsf{thrtns}(o_1, q(\vec{u}))$*, or*
3. *there exists an* o_3 *and* $q(\vec{v})$ s.t. $\langle o_3, q(\vec{v}), o_1, q(\vec{u}) \rangle \in V_P$, $o_1 \prec o_2$ *and* $\mathsf{thrtns}(o_2, q(\vec{u}))$.

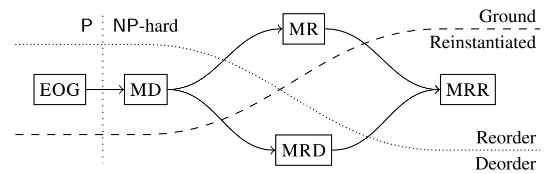

Figure 3: POP optimisation hierarchy. Arrows lead from less to more general optimisations.

While EOG cannot guarantee even a minimal deorder of its input [Bäckström, 1998], it can be computed in polynomial time and in practice consistently finds minimum deorderings [Muise *et al.*, 2016]. It thus serves an effective baseline for evaluating the MAXSAT-based optimisation techniques.

Figure 3 depicts the hierarchy of (the problems solved by) these algorithms. EOG is in P, all others are NP-hard to compute optimally. MD (*minimum deorder*) searches for a minimally ordered POP but disallows reorderings and changes to variable bindings. This is relaxed by both MR (*minimum reorder*), which allows reorderings, and MRD (*minimum reinstantiated deorder*), which allows bindings to change. The most general is MRR (*minimum reinstantiated reorder*), which allows both reorderings and changes to bindings.

Test cases (i.e., input plans) were generated by giving all first-order IPC STRIPS planning instances to three planners. To ensure a variety of plans, three planners of distinct "types" were used: the novelty-driven best-first search planner Dual-BFWS [Lipovetzky and Geffner, 2017], the heuristic forward-search planner LAMA [Richter and Westphal, 2010] and the SAT planner Madagascar [Rintanen, 2010]. Each (unique) plan was encoded with each encoding[2], and the resulting MAXSAT instances were preprocessed with MaxPre [Korhonen *et al.*, 2017] and given to the Loandra MAXSAT solver. Plan generation and encoding/optimisation were both limited to 8GB and 30m at 2.60GHz.

Because the optimality criteria in Definitions 9 and 11 cannot compare POPs with different operators, POP flexibilty is here assessed with the commonly-used *flex* measure [Nguyen and Kambhampati, 2001; Muise *et al.*, 2016; Siddiqui and Haslum, 2012], which is computed from the proportion of operators that are not (transitively) ordered:

Definition 20. *If* $P = \langle O, \theta, \prec \rangle$ *is a POP then* $\text{flex}(P) \stackrel{\text{def}}{=} 1 - (|\prec| / \sum\{i : 1 \leq i < |\, O \,|\})$.

A POP's *flex* ranges from 0 to 1 (a totally-ordered and unordered POP, resp.) and is a strong indicator of the number of linearisations it represents [Muise *et al.*, 2016].

5.2 Results

Results are summarised in Table 1. Domains where no plan was improved by any technique (*agricola*, *pegsol*, *snake*, *organic-synth-split*, *sokoban*, *visitall*) have been excluded.

[2]Implementation is available at bitbucket.org/max_waters/mrr

		MR			MRD			MRR		
Domain	f_{EOG}	T	C	Δ_{EOG}	T	C	Δ_{EOG}	T	C	Δ_{EOG}
airport (115)	0.25	0.13	63%	30%	1.28	28%	0%	1.9	24%	30%
barman (74)	0.01	30	41%		2.75	77%		28.79	36%	
childsnack (90)	0.69	15.77	100%	4%	20.64	100%	6%	29.99	99%	6%
data (22)	0.36	6.93	86%		25.53	50%		30	14%	
depot (63)	0.38	9.85	90%	8%	15.02	71%	28%	26.42	44%	44%
driverlog (59)	0.37	4	97%	2%	7.35	78%	17%	15.9	73%	34%
elevators (101)	0.43	3.71	65%		6.28	59%	12%	24.09	56%	28%
floortile (65)	0.39	20.2	100%		16.58	98%		30	5%	
freecell (147)	0.14	1.08	100%	2%	7.07	95%	40%	21.59	56%	**94%**
ged (91)	0.11	9.6	92%		9.7	65%	0%	10.21	60%	
gripper (60)	0.02	26.33	85%	19%	0.39	100%	0%	25.15	42%	29%
hiking (82)	0.13	4.03	94%	6%	8.08	96%	22%	20.96	44%	42%
logistics98 (101)	0.64	5.34	67%	3%	20.55	45%	8%	26.92	28%	9%
mprime (97)	0.22	0.15	100%		13.11	77%	54%	14.1	78%	60%
mystery (51)	0.22	0.06	100%		4.92	96%	44%	8.12	94%	48%
nomystery (95)	0.04	2.72	100%	12%	9.22	96%		17.88	60%	39%
parcprinter (118)	0.7	0.12	100%	0%	14.63	100%		24.95	99%	
parking (80)	0	0.86	100%	0%	26.73	35%		29.32	5%	
pipesworld (89)	0.19	0.85	100%	2%	7.42	91%	74%	21.99	62%	**96%**
rovers (120)	0.67	9.82	83%	1%	16	61%	6%	23.8	49%	12%
satellite (100)	0.5	0.94	88%		21.39	66%	4%	25.76	40%	8%
scanalyzer (104)	0.37	6.53	97%	8%	20.67	80%	35%	24.97	53%	38%
termes (22)		30	5%		8.44	27%		30	0%	
tetris (79)	0.63	0.78	96%		20.67	49%	6%	25.16	32%	9%
thoughtful (40)	0.14	5.54	100%		12.21	100%		22.66	33%	
tpp (89)	0.34	13.66	66%	13%	19.46	52%	32%	23.11	43%	49%
transport (98)	0.34	8.44	70%		18.21	58%	37%	26.84	46%	46%
w-working (120)	0.88	0.1	100%	0%	16.65	86%	3%	20.92	73%	3%
zenotravel (57)	0.41	2.03	95%	1%	11.63	77%	17%	20.54	65%	21%
ALL (2429)	0.4	6.06	87%	2%	13.52	74%	13%	22.43	52%	20%

Table 1: For each domain, f_{EOG} is the mean *flex* produced by EOG, and for each encoder, C is the % of plans for which a (satisficing or optimal) solution was found, T is the mean run time in minutes, and Δ_{EOG} is the mean % *flex* increase over EOG. All *flex* values are computed from plans for which every encoder found a solution, and all Δ_{EOG} values are statistically significant with $p < 0.01$ as calculated from a single-tailed, paired t-test. Empty cells indicate no data, or no significant difference.

Effectiveness of EOG

Our results confirm those of Muise *et al.* [2016] that EOG *always finds a minimum deorder of the input plan*, despite a lack of optimality guarantees. Also, EOG takes $< 3s$ in 88% of plans, with nearly 100% coverage. We thus exclude MD from the rest of the discussion.

As well as consistently finding minimum deorders, EOG found a minimum reorder in 69% of plans the plans that were solved optimally by MR, a minimum reinstantiated deorder in 53% of the plans solved optimally by MRD, and a minimum reinstantiated reorder in 45% of the plans solved optimally by MRR. As EOG can neither reorder actions nor rebind variables, this means that in a significant number of plans, *no reordering or reinstantiation was necessary in order to find an optimally ordered POP*.

Benefit of Reinstantiation

To assess the practical benefit of reinstantiation, we compare the coverage, run time and final *flex* of MRD and MRR with EOG and MR. To allow a meaningful comparison, mean *flex* values are computed from plans for which a solution was found by all encoders.

Overall, MRR provides a 20% *flex* increase over EOG.[3] This contrasts with the 3% *flex* increase provided by MR, meaning that reinstantiation has yielded a further *flex* increase of 17%. This improvement is consistent: MRR times out before finding as flexible a solution as EOG in $<$ 1% of the 1260 plans solved by both, with similar results for MR.

MRR improves on EOG in 24 domains. The largest differences are in *freecell* and *pipesworld*, where MRR provides *flex* increases of 94% and 96%, resp., while MR provides 2%, and *mprime* and *transport*, where MRR provides increases of 60% and 46%, resp., and MR provides no significant benefit.

However, this additional flexibility comes at significant computational cost. MRR has a mean coverage and run time of 52% and 22.43m, resp., with coverage dropping significantly in *floortile* (5%), *parking* (5%) and *termes* (0%).

The less general MRD approach provides less additional *flex* at less computational cost. With a mean coverage and run time of 74% and 13.52m, resp., it provides an overall *flex* increase of 13% over EOG and statistically significant increases in 21 domains, *pipesworld* being the highest (74%).

Interestingly, the decrease in coverage is not simply due to larger MAXSAT formulae. Table 2 shows that, as expected, MRD and MRR generate the largest encodings, and coverage always decreases with encoding size. However, within each size range, the encoders' coverages differ significantly (e.g., MRD and MR always have more coverage than MRR), suggesting that the problem is the formulae's structure, not size. A likely cause is so-called *object symmetries*: combinations of differently labelled but functionally equivalent domain objects (e.g., rovers R_1 and R_2 in Figure 1). Their presence can result in a small increase in formula size but an exponential number of equivalent (non-) solutions.

	MR		MRD		MRR	
Encoding Size	n_P	C	n_P	C	n_P	C
0 — $10.0 \cdot 10^4$	1,197	100%	358	100%	268	100%
$10.0 \cdot 10^4$ — $2.0 \cdot 10^5$	210	100%	221	100%	180	99%
$2.0 \cdot 10^5$ — $4.0 \cdot 10^5$	181	100%	241	100%	213	94%
$4.0 \cdot 10^5$ — $8.0 \cdot 10^5$	154	99%	265	95%	248	85%
$8.0 \cdot 10^5$ — $1.6 \cdot 10^6$	167	99%	250	89%	259	68%
$1.6 \cdot 10^6$ — $3.2 \cdot 10^6$	112	85%	269	82%	241	44%
$3.2 \cdot 10^6$ — $6.4 \cdot 10^6$	82	30%	202	57%	220	27%
$6.4 \cdot 10^6$ — $1.3 \cdot 10^7$	33	15%	211	50%	229	14%
$1.3 \cdot 10^7$ — $2.6 \cdot 10^7$	6	0%	119	38%	206	15%
$2.6 \cdot 10^7 +$	1	0%	7	14%	79	0%

Table 2: Plan count (n_P) and coverage (C) by encoding size (in number of clauses) for MR, MRD and MRR. Note that the encoding size is a log scale, each size range being twice that of the previous.

Influence of Planner

Finally, we note that the source of the input plan influences plan flexibility. Madagascar produces POPs, more specifically parallel plans where each step is a set of unordered actions. Interestingly, EOG can still increase their initial mean *flex* of 0.08 to 0.5 by removing unnecesary ordering constraints produced by the division of the plan into steps. MRR further improves on this by 15%. Over the sequential (*flex* = 0) plans generated by Dual-BFWS and LAMA, EOG gives a mean *flex* of 0.38 and 0.36, resp., which is improved upon by MRR by 30% and 14%, resp. Thus, while Dual-BFWS benefits most from reinstantiation, optimised Madagascar plans are the most flexible.

[3] All stated *flex* differences are statistically significant with $p < 0.01$ as calculated from a single-tailed, paired t-test.

6 Discussion

This paper presented a practical technique for plan optimisation through the modification of both ordering and variable binding constraints. Results show that in 52% of cases, MRR provides a *flex* increase of 20% over the EOG baseline in 22m, with results varying by domain. However, as reflected by the low coverage of MRR, this comes with considerable commputational cost. Additionally, in a significant number of cases, the additional computation simply reveals that the original bindings were already optimal.

Whether MRR is preferable to less costly methods such as EOG depends on the application. If offline preprocessing time is available and execution-time flexibilty is paramount, or the application domain is one where reinstantiation can quickly and consistently improve plan flexibility (e.g., *mystery*), then reinstantiation is clearly worthwhile.

This work generalises the POP optimality definitions of Bäckström [1998] and optimisation techniques of Muise *et al.* [2016] to allow changes in variable bindings. Other POP optimisation techniques have been studied. Muise *et al.* present extensions to MD and MR that also minimise the POP's size by removing any actions without dependent consumers (a problem shown to be NP-complete by Olz and Bercher [2019]), and Bercher and Olz [2020] study the theoretical aspects of modifying a POP's orderings in order to minimise its makespan. Siddiqui and Haslum [2012]'s *block decomposition* partitions a POP's actions into macro-operator *blocks*. The implicit requirement that blocks not be interleaved means that this approach can sometimes deorder plans that cannot be deordered by standard methods. Say *et al.* [2016] present MILP models for optimising POP size and flexibility. They avoid the cubic size of Formula 2 by using variables to represent action start and end times, resulting in significant efficiency gains over MD and MR.

Unlike the work presented here, the above techniques do not allow variable bindings to change. Thus, further work could investigate whether generalising these techniqes by allowing reinstantiation results in more flexible block decompositions or further reductions in plan size or makespan, and whether a MILP encoding of MRD and MRR can provide an improvement in execution time.

More general optimisation approaches relax a plan into a *partial plan*, which need not completely specify orderings *or bindings*. While a POP's linearisations are different orderings of the same set of actions, a partial plan defines of a set of classical plans that differ in both their ordering and variable bindings. Kambhampati and Kedar [1994] generalise EOG, and keep only the ordering and binding constraints needed to enforce POCL validity. Waters *et al.* [2018] show that partial plan constraints are intractable, limiting their execution-time use, and so present an algorithm that searches for partial plans that can be instantiated in polynomial time.

In Section 5, we suggest that object symmetries prevent effective reinstantiation. As there is much work on symmetry in planning [Joslin and Roy, 1997; Riddle *et al.*, 2016; Fox and Long, 1999; Rintanen, 2003; Pochter *et al.*, 2011], symmetry breaking in the context of plan optimisation is an interesting area for future work.

References

[Aghighi and Bäckström, 2017] Meysam Aghighi and Christer Bäckström. Plan reordering and parallel execution - A parameterized complexity view. In *Proc. of AAAI*, pages 3540–3546, 2017.

[Bäckström, 1998] Christer Bäckström. Computational aspects of reordering plans. *Journal of Artificial Intelligence Research*, 9:99–137, 1998.

[Bercher and Olz, 2020] Pascal Bercher and Conny Olz. POP $\equiv$ POCL, right? Complexity results for partial order (causal link) makespan minimization. In *Proc. of AAAI*, 2020.

[Berg *et al.*, 2019] Jeremias Berg, Emir Demirovic, and Peter J. Stuckey. Core-boosted linear search for incomplete MaxSAT. In *Proc. of the 16th International Conference on the Integration of Constraint Programming, AI, and Operations Research*, pages 39–56, 2019.

[Brachman and Levesque, 2004] Ronald Brachman and Hector Levesque. *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers, USA, 2004.

[Fox and Long, 1999] Maria Fox and Derek Long. The detection and exploitation of symmetry in planning problems. In *Proc. of IJCAI*, pages 956–961, 1999.

[Joslin and Roy, 1997] David Joslin and Amitabha Roy. Exploiting symmetry in lifted CSPs. In *Proc.of AAAI*, page 197–202, 1997.

[Kambhampati and Kedar, 1994] Subbarao Kambhampati and Smadar Kedar. A unified framework for explanation-based generalization of partially ordered and partially instantiated plans. *Artificial Intelligence*, 67(1):29–70, 1994.

[Kambhampati and Nau, 1996] Subbarao Kambhampati and Dana S. Nau. On the nature and role of modal truth criteria in planning. *Artifical Intelligence*, 82(1-2):129–155, 1996.

[Korhonen *et al.*, 2017] Tuukka Korhonen, Jeremias Berg, Paul Saikko, and Matti Järvisalo. MaxPre: An extended MaxSAT preprocessor. In *Proc. of the 20th Int. Conference on Theory and Applications of Satisfiability Testing*, volume 10491 of *LNCS*, pages 449–456. Springer, 2017.

[Lifschitz, 1987] Vladimir Lifschitz. On the semantics of STRIPS. In Michael Georgeff and Amy Lansky, editors, *Reasoning about Actions and Plans*, pages 1–9. Morgan Kaufmann, San Mateo, CA, 1987.

[Lipovetzky and Geffner, 2017] Nir Lipovetzky and Hector Geffner. Best-first width search: Exploration and exploitation in classical planning. In *Proc. of AAAI*, pages 3590–3596, 2017.

[Muise *et al.*, 2016] Christian Muise, J. Christopher Beck, and Sheila A. McIlraith. Optimal partial-order plan relaxation via MaxSAT. *Journal of Artificial Intelligence Research*, 57:113 – 149, 2016.

[Nguyen and Kambhampati, 2001] XuanLong Nguyen and Subbarao Kambhampati. Reviving partial order planning. In *Proc. of IJCAI*, pages 459–464, 2001.

[Olz and Bercher, 2019] Conny Olz and Pascal Bercher. Eliminating redundant actions in partially ordered plans - A complexity analysis. In *Proc. of ICAPS*, pages 310–319, 2019.

[Pochter *et al.*, 2011] Nir Pochter, Aviv Zohar, and Jeffrey S. Rosenschein. Exploiting problem symmetries in state-based planners. In *Proc. of AAAI*, 2011.

[Richter and Westphal, 2010] Silvia Richter and Matthias Westphal. The LAMA planner: Guiding cost-based anytime planning with landmarks. *Journal of Artificial Intelligence Research*, 39(1):127–177, 2010.

[Riddle *et al.*, 2016] Pat Riddle, Jordan Douglas, Mike Barley, and Santiago Franco. Improving performance by reformulating PDDL into a bagged representation. In *Proc. of the 8th Workshop on Heuristics and Search for Domain-independent Planning*, pages 28–36, 2016.

[Rintanen, 2003] Jussi Rintanen. Symmetry reduction for SAT representations of transition systems. In Enrico Giunchiglia, Nicola Muscettola, and Dana S. Nau, editors, *Proc. of ICAPS*, pages 32–40, 2003.

[Rintanen, 2010] Jussi Rintanen. Heuristics for planning with SAT. In David Cohen, editor, *Proc. of CP*, pages 414–428, 2010.

[Say *et al.*, 2016] Buser Say, André A. Ciré, and J. Christopher Beck. Mathematical programming models for optimizing partial-order plan flexibility. In *Proc. of ECAI*, pages 1044–1052, 2016.

[Shlyakhter, 2007] Ilya Shlyakhter. Generating effective symmetry-breaking predicates for search problems. *Discrete Applied Mathematics*, 155(12):1539–1548, 2007.

[Siddiqui and Haslum, 2012] Fazlul Hasan Siddiqui and Patrik Haslum. Block-structured plan deordering. In *Proc. of the 25th Australasian Joint Conference on Advances in Artificial Intelligence*, pages 803–814, 2012.

[Waters *et al.*, 2018] Max Waters, Bernhard Nebel, Lin Padgham, and Sebastian Sardiña. Plan relaxation via action debinding and deordering. In *Proc. of ICAPS*, pages 278–287, 2018.

[Weld, 1994] Daniel S. Weld. An introduction to least commitment planning. *AI Magazine*, 15(4):27–61, 1994.