Are White Knights Worth the Trouble? Reconciling POCL and Partial-Order Plans for Plan Optimization

Harrison Oates , Pascal Bercher
School of Computing, The Australian National University
harrison.oates@anu.edu.au, pascal.bercher@anu.edu.au

Abstract

Partial order causal link (POCL) planning offers rich structural representations for plan optimization. However, under the classical threat definition, POCL plans form a strict subset of valid partial order (PO) plans. This gap limits existing POCL optimization techniques to a restricted subspace of PO solutions. We revisit 'white knight' threat semantics, which permit a causal link to persist across a threat provided the deleted condition is re-established by an intermediate producer. We prove that under this definition, the POCL and PO solution spaces become equivalent. By extending a MaxSAT-based optimization framework, we demonstrate that white knights are theoretically necessary for completeness and practically advantageous: they yield plans with significantly fewer ordering constraints in domains with complex causal interference, while maintaining computational tractability on standard benchmarks.

1 Introduction

Plan optimization – finding plans that minimize makespan, cost, or other quality metrics – is a central problem in automated planning. Partial order (PO) planning is highly attractive here because it avoids unnecessary total orderings, exposing flexibility that can be exploited during both optimization and execution [Bercher *et al.*, 2024]. This flexibility can yield strictly higher-quality plans than alternative representations [Oates and Bercher, 2026b]. Among PO representations, partial order causal link (POCL) plans [McAllester and Rosenblitt, 1991; Penberthy and Weld, 1992] have proven especially effective. By explicitly identifying which actions support which preconditions, causal links clarify *why* particular ordering constraints exist [Bercher *et al.*, 2024]. Consequently, despite the computational hardness of reasoning over POCL structures [Bercher, 2021; Oates and Bercher, 2026c], they remain integral to state-of-the-art hierarchical planning [Bit-Monnot *et al.*, 2020; Bit-Monnot, 2023], temporal planning [Bit-Monnot and Godet, 2025], and plan optimization systems [Bercher *et al.*, 2024].

Despite often being treated interchangeably, PO plans and POCL plans are not equivalent under standard semantics. While every POCL plan induces a valid PO plan, the converse is false: some valid PO plans cannot be represented as threat-free POCL plans without introducing additional ordering constraints [Bercher and Olz, 2020]. As a result, techniques optimizing POCL plans search a strict subset of the PO solution space. This mismatch means that output plans may be more constrained and less flexible than necessary.

The cause of this gap is the classical definition of causal threats. Under standard semantics, if an action deletes a fluent protected by a causal link, the threat must be resolved via promotion (ordering the consumer before the deleter) or demotion (ordering the deleter before the producer). Both strategies force the threatening action outside the protection interval. However, this ignores a third possibility: the threatened fluent could be re-established by an intermediate action before it is needed by the consumer. Such an action, commonly called a *white knight*, ensures the plan remains executable without requiring promotion or demotion.

White knights are not a new idea: they appear in early work on partial order planning [Chapman, 1987], where they can be used to verify that a PO plan is a solution to a planning problem in polynomial time [Kambhampati and Nau, 1996]. More recently, a POCL threat definition incorporating white knights was used by Siddiqui and Haslum [2015] for plan quality optimization through block deordering, and the same definition was used in a textbook on the Planning Domain Definition Language [Haslum *et al.*, 2019, p. 41]. However, the semantic implications of white knights have not been fully recognized; in particular, their role in reconciling PO and POCL plan representations has remained unexplored.

In this paper, we show that adopting a white-knight-aware threat definition eliminates the representational gap between PO and POCL plans. We prove that under these refined threat semantics, the sets of valid PO plans and valid POCL plans coincide: every PO plan admits a threat-free POCL plan with exactly the same ordering constraints, and vice versa. This resolves the limitation identified by Bercher and Olz [2020] and clarifies the precise conditions under which causal-link reasoning can be applied without loss of generality.

This equivalence has immediate practical applications for plan optimization. In particular, we revisit the MaxSAT-based framework of Muise *et al.* [2016], which aims to optimize partial order plans but, due to its reliance on the standard threat definition, in fact optimizes only over standard

POCL plans. By incorporating white knight semantics into the encoding, we restore alignment between the optimization framework and its intended PO-plan solution space.

The remainder of this paper is structured as follows. Section 2 formalizes classical, PO, and POCL planning. Section 3 introduces white knight threat semantics, proves the equivalence of the solution spaces (Theorem 1), and details the corrected MaxSAT optimization encoding. Section 4 presents our empirical analysis, demonstrating both the structural necessity of white knights on synthetic domains and their minimal overhead on standard benchmarks. Finally, Section 5 summarizes our contributions.

2 Formal Framework

In this section, we provide a formal framework for classical planning, sequential plans, partial order (PO) plans, and partial order causal link (POCL) plans. We take care to build up these concepts systematically, as the distinction between PO and POCL plans is crucial to our main contributions. We base our work on a ground, i.e., *propositional* formalism, as solution plans are necessarily ground and this choice simplifies our presentation considerably.

A classical planning problem is a tuple $(\mathcal{F}, \mathcal{A}, \mathcal{I}, \mathcal{G})$ where $\mathcal{F}$ is a finite set of fluents, $\mathcal{I} \in 2^{\mathcal{F}}$ is the initial state, $\mathcal{G} \subseteq \mathcal{F}$ is the goal description, and $\mathcal{A}$ is the finite set of *actions*. A *state* is any $s \in 2^{\mathcal{F}}$.

An action $a \in \mathcal{A}$ is a tuple $a = (pre(a), add(a), del(a))$, where $pre(a) \subseteq \mathcal{F}$ denotes its preconditions, $add(a) \subseteq \mathcal{F}$ its add effects, and $del(a) \subseteq \mathcal{F}$ its delete effects. An action a is *applicable* (or executable) in state s if and only if $pre(a) \subseteq s$. The resulting state after applying a is given by the transition function $\gamma(a, s) = (s \setminus del(a)) \cup add(a)$. The most common plan representation is the sequential plan, which specifies a complete linear ordering of actions. An action sequence $\bar{a} = a_1, a_2, \ldots, a_n$ is applicable in a state s_0 if and only if there exists a sequence of states $s_0, s_1, \ldots, s_n$ such that for all $i \in [1, n]$, it holds that a_i is applicable in state s_{i-1} and generates state $s_i = \gamma(a_i, s_{i-1})$. We call s_n the state *generated* by $\bar{a}$.

Definition 1. *An action sequence $\bar{a}$ is called a sequential plan for a classical planning problem if and only if it is applicable in the initial state $\mathcal{I}$ and generates a goal state $s \supseteq \mathcal{G}$.*

While sequential plans are intuitive, they impose unnecessary constraints by specifying a total ordering over all actions. In many cases, certain actions are independent and could be executed in any relative order without affecting the plan's validity. Generalizing from a total order to a partial order allows us to represent this flexibility explicitly, which proves valuable both for plan execution and for reasoning about plan structure.

A partial PO plan is a tuple $(PS, \prec)$ consisting of a finite set of plan steps PS, with each step $(l, a) \in PS$ consisting of an action a with unique label l. The labelling is necessary to differentiate between multiple occurrences of the same action within a plan. The relation $\prec \subseteq PS \times PS$ is a strict partial order over the plan steps. A total ordering of the actions in PS that respects $\prec$ is a *linearization*.

Definition 2. *A partial PO plan $P = (PS, \prec)$ is called a PO plan for a classical planning problem if and only if every linearization is executable in the initial state $\mathcal{I}$ and generates a goal state $s \supseteq \mathcal{G}$.*

While PO plans elegantly represent ordering flexibility, they lack explicit representation of *why* particular orderings are necessary to guarantee executability. POCL plans address this limitation by making causal dependencies explicit through the addition of causal links, which annotate precisely which action produces which fluent for consumption by which other action.

A partial POCL plan is a tuple $(PS, \prec, CL)$ with PS and $\prec$ defined the same as in PO plans, and $CL \subseteq PS \times \mathcal{F} \times PS$ a finite set of causal links. In POCL plans, the initial and goal states are encoded as special plan steps, $init = (init, (\emptyset, \mathcal{I}, \emptyset))$ and $goal = (goal, (\mathcal{G}, \emptyset, \emptyset))$. $init$ is ordered strictly before all other plan steps, while $goal$ is ordered strictly after all plan steps. As we shall shortly see, this is necessary to obtain a coherent plan definition.

A causal link $(ps, f, ps') \in CL$ indicates that the fact f is *produced* by the plan step ps and *consumed* by the plan step ps'. This implies that $f \in pre(ps') \cap add(ps)$. To ease definitions, we require that a causal link $(ps, f, ps') \in CL$ implies an ordering constraint $(ps, ps') \in \prec$. In such a causal link, we call f *protected* by that link.

In standard POCL planning, we define a causal threat in the following manner.

Definition 3 (Causal Threat)**.** *A plan step ps'' threatens a causal link (ps, f, ps') if and only if $f \in del(ps'')$ and the transitive closure of $\prec \cup \{(ps, ps''), (ps'', ps')\}$ is a strict partial order.*

Causal threats are resolved by imposing additional ordering constraints on the threatening step. For a causal link (ps_p, f, ps_c) threatened by the plan step ps_t, *promotion* enforces an ordering $ps_c \prec ps_t$, preventing ps_t from deleting f before it can be used by ps_c. Similarly, *demotion* enforces an ordering constraint $ps_t \prec ps_p$ so ps_t cannot execute over the interval. We can now define valid POCL plans.

Definition 4. *A partial POCL plan $(PS, \prec, CL)$ is a POCL plan for a classical planning problem if and only if every precondition is protected by a causal link and there are no causal threats.*

This definition implies that every linearization of a POCL plan is executable, and so every POCL plan induces a valid PO plan simply by discarding all causal links. However, the converse is known not to hold under the standard definitions [McAllester and Rosenblitt, 1991; Kambhampati and Nau, 1996; Bercher and Olz, 2020]. To show this, consider the partial POCL plan in Figure 1. Straightforwardly, if we remove its causal links, it is a PO plan as all six of its linearizations are executable and end up in a goal state. However, adding a causal link from w_1 or w_2 to support the open precondition P will raise a causal threat with s_2 or s_1, respectively. Resolving the threat in each case requires promotion or demotion, thereby adding ordering constraints to the induced PO plan and reducing the set of valid linearizations.

Consequently, 'partial order' plan optimization techniques which use causal links, such as the MaxSAT encoding of Muise *et al.* [2016], actually optimize over the space of POCL

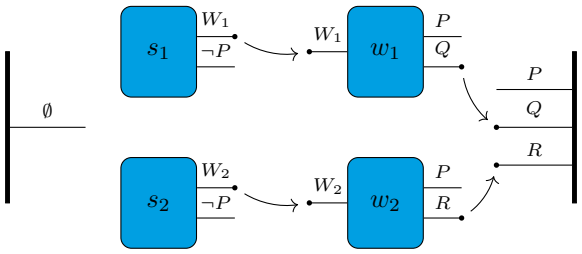

Figure 1: There are PO plans for which there is no POCL plan with the same set of ordering constraints. Example due to Kambhampati, as cited in McAllester and Rosenblitt [1991].

plans, rather than over PO plans as claimed. In the next section, we discuss how a slight adjustment to the causal threat definition makes PO and POCL plans equivalent, thus allowing MaxSAT and other POCL-oriented techniques to be applicable to PO plan optimization.

3 White Knights To The Rescue

In fairy tales, a white knight saves the kingdom. In planning, the stakes are slightly lower, but the mechanism is the same: a *white knight* rescues a once-established and then deleted fluent, reasserting a necessary condition just in time to save the plan from failure [Chapman, 1987]. Siddiqui and Haslum [2015, Def. 1] and Haslum *et al.* [2019, p. 41] use a formalization that can be regarded as a hybrid between the standard POCL plan and PO plan formalizations we introduced in the previous section. In this formalization, causal links are used, but the definition of causal threat now depends on white knights.

Definition 5 (White Knight)**.** *Let (ps_p, f, ps_c) be a causal link and ps_{del} be a plan step such that $f \in del(ps_{del})$. A plan step ps_{wk} is a* white knight *for this link with respect to ps_{del} if and only if: (1) $f \in add(ps_{wk})$, and (2) the ordering constraints enforce $ps_{del} \prec ps_{wk} \prec ps_c$.*

Definition 6 (White Knight Causal Threat)**.** *A causal link (ps_p, f, ps_c) is* threatened *under white knight semantics if and only if there exists a plan step ps_{del} with $f \in del(ps_{del})$ such that: (1) The step is a standard causal threat (i.e., it is not the case that $ps_{del} \prec ps_p$ or $ps_c \prec ps_{del}$), and (2) There exists no white knight ps_{wk} for the link with respect to ps_{del}.*

With this refined threat definition, we can now define valid POCL plans under the white knight semantics.

Definition 7 (wk-POCL plans)**.** *A partial POCL plan $P = (PS, \prec, CL)$ is a* wk-POCL plan *for a classical problem if and only if every precondition of every step in PS is protected by a causal link in CL, and there are no white knight causal threats.*

To see how this definition resolves the gap identified in the introduction, consider again the plan structure in Figure 1. Suppose we satisfy the goal's precondition P with a causal link from w_1, i.e., $(w_1, P, goal) \in CL$. Under standard semantics, step s_2 threatens this link because it deletes P and is unordered with respect to the protection interval $w_1 \rightarrow goal$. However, under Definition 7, we evaluate whether s_2 constitutes a *white knight causal threat*. We identify step w_2 as a white knight because $P \in add(w_2)$ and the orderings $s_2 \prec w_2 \prec goal$ hold (enforced by the link (s_2, W_2, w_2) and the goal constraint). Since a valid white knight exists, the threat from s_2 is resolved. Symmetrically, if P were supported by w_2, step w_1 would act as the white knight for the threat from s_1. Thus, the plan is a valid wk-POCL plan without the spurious ordering constraints (e.g., $s_2 \prec w_1$) required by the standard definition. With white knight semantics introduced, we can now state our main theoretical result.

Theorem 1. *Let Π be a planning problem. $(PS, \prec)$ is a valid PO plan for Π if and only if there exists a set of causal links CL such that $(PS, \prec, CL)$ is a valid wk-POCL plan for Π.*

Proof. $\Leftarrow$. Let P be a wk-POCL plan. It is straightforward to see that Definition 7 implies every linearization of P is executable, and so P is a valid PO plan.

$\Rightarrow$. Let $P = (PS, \prec)$ be a valid PO plan. We construct a wk-POCL plan $P' = (PS, \prec, CL)$ from P by adding causal links, and prove that there are no causal threats.

Construction of causal links: For each plan step $ps_c \in PS$ and each precondition $f \in pre(ps_c)$, we must choose a producer ps_p such that $f \in add(ps_p)$ and $ps_p \prec ps_c$, then add the causal link (ps_p, f, ps_c) to CL.

To see that such a ps_p exists: Since P is valid, f holds at ps_c in every linearization, so some step must add f before ps_c in every linearization. Let $\mathcal{A}_f = \{ps \in PS : f \in add(ps)\}$ be the set of all steps that add f. If no $ps_p \in \mathcal{A}_f$ satisfies $ps_p \prec ps_c$, then for each producer we could construct a linearization placing ps_c before that producer. Since $\prec$ would not constrain any producer to precede ps_c, we could construct a linearization where ps_c appears before all producers of f, meaning f would not hold at ps_c, contradicting validity. Hence some $ps_p \in \mathcal{A}_f$ satisfies $ps_p \prec ps_c$, and we select any such ps_p for the causal link.

No causal link is threatened: We now prove that for any causal link $(ps_p, f, ps_c) \in CL$, the link is not threatened under white knight semantics.

Consider any plan step ps_d such that $f \in del(ps_d)$. We must show that at least one of the following holds:

1. $ps_c \prec ps_d$, or
2. There exists ps_{wk} such that $f \in add(ps_{wk})$ and $ps_d \prec ps_{wk} \prec ps_c$.

Suppose condition (1) does not hold, i.e., $ps_c \not\prec ps_d$. We prove condition (2) must hold by contradiction. Assume condition (2) also fails. Then for every step $ps' \in PS$ with $f \in add(ps')$, we have $\neg(ps_d \prec ps' \prec ps_c)$. This means each producer of f falls into one of the following categories:

- $ps' \prec ps_d$: the producer is constrained to appear before the deleter, or
- $ps_c \prec ps'$: the producer is constrained to appear after the consumer, or
- The producer is not constrained by $\prec$ to appear strictly between ps_d and ps_c.

We now construct a linearization of P in which f does not hold at ps_c, contradicting the validity of P.

Since $ps_c \not\prec ps_d$, the ordering ps_d before ps_c is consistent with $\prec$. We build a linearization π^* that respects $\prec$, places ps_d before ps_c, and for each producer $ps' \in \mathcal{A}_f$:

- If $ps' \prec ps_d$, then ps' appears before ps_d in π^*.
- If $ps_c \prec ps'$, then ps' appears after ps_c in π^*.
- Otherwise, since $\neg(ps_d \prec ps' \prec ps_c)$, we can place ps' either before ps_d or after ps_c without violating $\prec$. We choose one such placement.

By construction, π^* respects all constraints in $\prec$. In π^*:

- Every step that adds f appears either before ps_d or after ps_c.
- The step ps_d deletes f and appears before ps_c.
- No step re-establishes f between ps_d and ps_c.

Therefore, f does not hold when ps_c executes in π^*. But π^* is a valid linearization of P, contradicting the assumption that P is a valid PO plan. Hence condition (2) must hold.

Since for every deleter ps_d at least one of conditions (1) or (2) holds, the causal link (ps_p, f, ps_c) is not threatened. Since every precondition in P' is supported by a causal link, and we have shown that no such link is threatened under white knight semantics, P' satisfies Definition 7. Thus, every valid PO plan can be augmented into a valid wk-POCL plan. □

Having established that wk-POCL semantics capture the entire space of valid PO plans, we can now formally quantify their advantage over the standard formulation. The following result confirms that our definition is a strict generalization of the classical approach.

Theorem 2 (Strict Subsumption)**.** *Every valid POCL plan is a valid wk-POCL plan. Furthermore, the inclusion is strict: there exist valid wk-POCL plans $(PS, \prec, CL)$ for which there exists no valid standard POCL plan $(PS, \prec', CL)$ such that $\prec' \subseteq \prec$.*

Proof. The first claim is trivial by definition: standard semantics resolve threats via promotion or demotion, which satisfies the first condition of the white knight threat definition (Definition 5). Thus, any plan valid under standard semantics is valid under wk-semantics. The second claim follows from the counter-example in Figure 1. As detailed above, this plan can be augmented into a valid wk-POCL plan with the depicted orderings $\prec$ by choosing either w_1 or w_2 to support the open precondition P. However, under standard semantics, the threat from s_2 (respectively, s_1) must be resolved by demotion. This yields a plan with orderings $\prec'$ where $\prec \subset \prec'$. □

3.1 Optimizing PO Plans Using MaxSAT

Having established the theoretical equivalence of wk-POCL plans with PO plans, we now show how to leverage white knights for plan optimization. We adopt the partial weighted MaxSAT framework of Muise *et al.* [2016]. For the sake of completeness, we reproduce their base encoding variables and constraints (Equations 1–7), which enforce standard POCL validity. Our specific contribution is the replacement of the standard threat resolution constraint with a white-knight-aware formulation, thereby enabling optimization over the full PO space.

The aim of our optimization efforts is to produce maximally flexible PO plans, in the sense that decisions regarding plan execution can be deferred. In this context, we use the number of ordering constraints in the *transitive closure* of the plan as a proxy for flexibility. Consequently, when we discuss minimizing 'ordering constraints', we refer to minimizing the cardinality of the strict partial order relation, $|\prec|$.

We first provide the necessary definition for the allowed plan modification [Bäckström, 1998].

Definition 8. *Let P and Q be wk-POCL plans for a problem Π, with $P = (PS, \prec, CL)$ and $Q = (PS', \prec', CL')$. Then Q is a reordering of P if and only if $PS = PS'$.*

Definition 9 (Minimum Reordering)**.** *Let P be a wk-POCL plan. A reordering Q of P is* minimum *if there exists no reordering Q' of P such that $|\prec'| < |\prec|$.*

Partial weighted MaxSAT is an optimization variant of the Boolean satisfiability problem where clauses are partitioned into hard clauses (which must be satisfied) and soft clauses (which we attempt to maximize the sum of the weights of the satisfied clauses). For plan optimization, hard clauses encode plan validity while soft clauses encode quality metrics.

Let $P = (PS, \prec, CL)$ be a valid wk-POCL plan. A satisfying assignment to the encoding $\Phi(P)$ induces a *target* wk-POCL plan. We use the following Boolean variables: x_{ps} indicates that the plan step ps appears in the target plan, $\kappa(ps_i, ps_j)$ indicates that $ps_i \prec ps_j$ in the target plan, and $\Upsilon(ps_p, f, ps_c)$ indicates that (ps_p, f, ps_c) is a causal link in the target plan.

The following hard clauses ensure that the induced ordering relation is a strict partial order and that all plan steps are properly embedded between $init$ and $goal$.

$$\neg\kappa(ps, ps) \tag{1}$$

$$x_{init} \wedge x_{goal} \tag{2}$$

$$\kappa(ps_i, ps_j) \rightarrow x_{ps_i} \wedge x_{ps_j} \tag{3}$$

$$x_{ps} \rightarrow \kappa(x_{init}, ps) \wedge \kappa(ps, x_{goal}), ps \notin \{x_{init}, x_{goal}\} \tag{4}$$

$$\kappa(ps_i, ps_j) \wedge \kappa(ps_j, ps_k) \rightarrow \kappa(ps_i, ps_k) \tag{5}$$

Under the POCL solution criterion, every included plan step must have each of its preconditions protected by a causal link.

$$x_{ps_j} \rightarrow \bigwedge_{f \in pre(ps_j)} \bigvee_{\substack{ps_i \in PS \\ : f \in add(ps_i)}} (\kappa(ps_i, ps_j) \wedge \Upsilon(ps_i, f, ps_j)) \tag{6}$$

The crucial difference from the original encoding is in the treatment of causal threats. Standard POCL allows for threats to be resolved through promotion and demotion, and so the constraint is defined accordingly:

$$\Upsilon(ps_p, f, ps_c) \rightarrow \bigwedge_{\substack{ps_d \in PS \\ : f \in del(ps_d)}} x_{ps_d} \rightarrow \left(\underbrace{\kappa(ps_d, ps_p) \vee \kappa(ps_c, ps_d)}_{\text{Threat resolution Clause}} \right). \tag{7}$$

Under wk-POCL semantics, we add a disjunct in the threat resolution clause to account for white knights:

$$\bigvee_{ps_{wk} \in PS : f \in add(ps_{wk})} (\kappa(ps_d, ps_{wk}) \wedge \kappa(ps_{wk}, ps_c)). \tag{8}$$

To define the necessary optimization metric, soft clauses $\neg\kappa(ps_i, ps_j)$ for all $ps_i, ps_j \in PS$ with weight 1 are used. In this paper, we focus on optimal reorderings to isolate the effects of white knights on plan flexibility, and so we require every plan step to be in the target plan, i.e., x_{ps_i} for all $ps_i \in PS$. However, Muise *et al.* [2016] define several additional optimization variants, including action elimination. Our extension to white-knight semantics leaves the objective function and step-selection constraints unchanged. Consequently, the wk-POCL encoding is compatible with all optimization variants they propose.

4 Experimental Evaluation

While a minimum reorder of a PO plan can lead to fewer ordering constraints and hence more flexibility regarding execution orders compared to a minimum POCL plan, it is unclear whether this occurs often in practice, and at what cost. To answer this question, we extended the MaxSAT POCL encoding library POPGEN to support wk-POCL plans.[1] We conducted a comparative analysis by computing the optimal reordering for each instance under both definitions. All experiments were run on a Linux environment equipped with an Intel Core i5-1240P CPU and 32GB of RAM. We utilized the RC2 MaxSAT solver [Ignatiev *et al.*, 2019] provided by the PySAT library. Experimental details and reproducibility data are available online [Oates and Bercher, 2026a].

4.1 Plan Generation and Setup

For the broad evaluation, we selected domains from the International Planning Competitions (IPC). To generate the initial sequential plans, we utilized the `planutils` toolkit [Muise *et al.*, 2022] to run the SCORPION MAIDU planner [Corrêa *et al.*, 2023], a joint winner in the IPC 2023 Satisficing Track. Initial plan generation was limited to 30 min and 8 GB RAM per instance, while plan optimization (encoding + solving) was limited to 30 min and 4 GB RAM per instance. We report coverage (instances solved), total optimization time, and the resulting plan quality. Our primary quality metric is the cardinality of the transitive closure ($|\prec|$), as this corresponds directly to the objective function of the MaxSAT encoding.

It is important to note that for a fixed number of ordering constraints (the optimization objective), there may exist multiple distinct partial orderings that yield varying numbers of linearizations. Since the optimizer solely targets constraint minimization, it is indifferent between two plans with the same number of constraints, even if one allows significantly more linearizations. Consequently, differences in linearization counts between plans of identical cost are often artifacts of solver tie-breaking. Therefore, we adopt a conservative criterion: we consider the wk-POCL encoding to have strictly outperformed the standard encoding only when it yields a plan with strictly fewer ordering constraints ($|\prec|_{wk} < |\prec|_{std}$).

4.2 Results on Standard Benchmarks

We first compare the performance of both encodings across a number of domains from previous International Planning Competitions. The goal of this experiment was to quantify the overhead of white knight semantics in scenarios where they may not be strictly necessary, and to verify that the encoding scales gracefully compared to the standard approach.

Table 1 summarizes domain-specific performance across solved instances. Across each instance, both coverage and the number of ordering constraints found by the wk-POCL encoding were identical to that of the standard encoding. That is, the wk-encoding did not find strictly more flexible plans than the standard encoding.

A potential concern with a more expansive threat definition is the inflation of the SAT instance size. We analyzed the size of the CNF encodings generated for all 81 valid instances. As shown in Table 2, the introduction of white knight semantics necessitates auxiliary variables to maintain a compact CNF representation. Empirically, this resulted in a mean increase of 16.1% in the number of Boolean variables. However, the impact on constraint complexity is minimal, with the number of clauses increasing by only 2.7% on average.

The runtime penalty for the added completeness is negligible. The average total time (encoding + solving) rose from 38.87s to 41.55s. A Wilcoxon signed-rank test [Wilcoxon, 1945] on the plan runtimes yields $p = 0.30$, indicating no statistically significant difference in performance. Similarly, separate tests for encoding time ($p \approx 0.27$) and solving time ($p \approx 0.13$) confirm that the wk-POCL encoding is computationally competitive with the incomplete standard encoding. As shown in Table 4 and Figure 2, the average problem actually benefits slightly from the modification. The median total overhead ratio is $0.990\times$, indicating that for the typical instance, the wk-encoding is marginally faster than the standard encoding. The discrepancy between the rise in aggregate average time and the neutral mean ratio ($0.998\times$) suggests that the cost of completeness is concentrated in specific outliers (Max Total Ratio $1.699\times$), rather than presenting a systematic slowdown across the benchmark set.

4.3 Synthetic Benchmark

The results on standard IPC benchmarks indicate that while the standard and white knight encodings yield plans with identical ordering constraints, their internal causal justifications frequently differ. As shown in Table 1, white knights were actively exploited to ensure plan validity in 40 of the 64 solved instances, with an average of 7.45 white knights among those instances in which they were used. The fact that this utilization did not result in fewer ordering constraints suggests that standard domains exhibit a high degree of structural rigidity: while the establish-delete-reestablish patterns exist and are used, other parallel causal dependencies enforce the same strict partial order, masking the potential flexibility gains of the white knight.

To strictly evaluate the benefit of the white knight definition, we constructed a synthetic "Resource Borrowing" domain that generalizes the counter-example from Figure 1. The problem consists of N agents and a single global fluent `(shared-resource)`. The goal requires `(shared-resource)` to be present at the end of the plan. Crucially, each agent must perform a task sequence that temporarily violates this goal condition: a *threat* action (deleting

[1] Code available at https://github.com/HarrisonOates/popgen.

		Coverage		Active wk		Avg. Total Time (s)		Avg. Constraints ($\lvert\prec\rvert$)		Runtime Increase (%)	
Domain	**Instances**	**Std**	wk	$\#/w \geq 1$	**Mean**	**Std**	wk	**Std**	wk	**Max**	**Mean**
Depots	11/22	9	9	6	8.67	38.1	39.6	27.8	27.8	11.99	-0.77
Driverlog	11/20	11	11	6	5.67	6.5	7.0	58.7	58.7	43.32	0.25
Rovers	17/20	10	10	4	11.00	200.5	215.5	54.3	54.3	1.71	-1.14
Storage	15/20	15	15	12	7.58	1.3	1.5	49.6	49.6	1.35	-5.45
TPP	15/30	8	8	4	9.25	4.0	4.0	26.5	26.5	12.08	-1.84
Zenotravel	12/20	11	11	8	5.00	1.5	1.5	23.9	23.9	69.86	5.49
Total	**81/132**	**64**	**64**	**40**	**7.45**	**38.9**	**41.6**	**207.6**	**207.6**	**69.86**	**-0.20**

Table 1: Comparison of coverage, runtime, and average ordering constraints across the evaluated IPC domains. Coverage indicates the number of instances where the MaxSAT solver found an optimal solution within the resource limits (30m, 4GB). 'Active wk' reports (i) the number of solved instances whose validity relied on at least one white knight causal link, and (ii) the average number of exploited white knights over those instances only. Runtimes include both encoding and solving. Note that the size of the transitive closure ($\lvert\prec\rvert$) is identical between encodings across all solved instances in these domains.

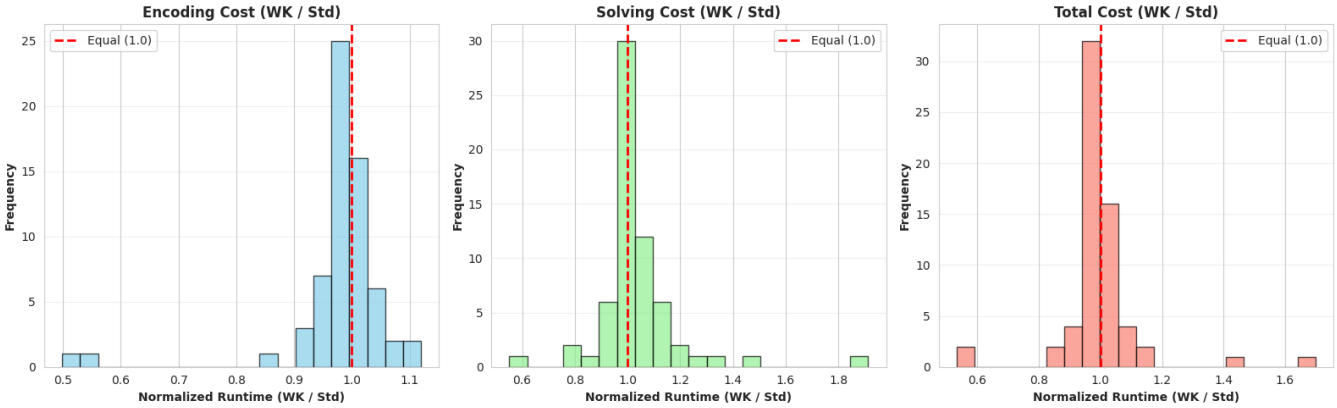

Figure 2: Performance overhead of white knight encoding on IPC problems. Ratios < 1.0 indicate the wk-encoding was faster.

	Variables		Clauses	
Metric	**Standard**	wk**-POCL**	**Standard**	wk**-POCL**
Mean	1,097	1,273	26,033	26,727
Median	632	718	7,386	8,164
Increase	**–**	**+16.1%**	**–**	**+2.7%**

Table 2: Comparison of the average number of variables and clauses generated by the Standard and wk-POCL encodings across all valid IPC instances.

Metric	**Standard**	wk**-POCL**
Coverage (Solved/Total)	64 / 81	64 / 81
Avg. Encoding Time	2.41s	2.30s
Avg. Solving Time	36.46s	39.24s
Avg. Total Time	38.87s	41.55s

Table 3: Summary of computational performance quality on IPC benchmarks ($N = 81$).

the resource) followed immediately by a *restore* action (re-adding it). Under standard POCL semantics, these threats force the imposition of ordering constraints to serialize the agents. Under wk-POCL, the *restore* action serves as a white knight, allowing the agents to operate in parallel.

We scaled the problem size from $N = 3$ to $N = 10$ agents. The results (Table 5) strongly validate our theoretical claims. The wk-POCL encoding consistently found plans with fewer ordering constraints than the standard encoding. On average, wk-POCL solutions contained 14.1% fewer ordering constraints. For the largest instance ($N = 10$), the standard encoding required 60 constraints to ensure validity, while the white knight encoding required only 51. As illustrated in Figure 3, this widening gap empirically confirms that the standard definition introduces spurious ordering constraints to resolve threats that are effectively benign under white knight semantics.

This reduction in constraints directly translates to increased plan flexibility. The wk-POCL plans admitted strictly more valid linearizations in every instance tested ($8/8$). The degree of improvement scales with problem complexity: at $N = 3$, wk-POCL offered a $1.36\times$ increase in valid linearizations; at $N = 10$, this factor grew to $2.14\times$.

This domain also serves as a "stress test" for the encoding size. Unlike the IPC domains where threats are sparse, the Resource Borrowing domain features dense causal interference: every agent's `Threaten` action constitutes a threat to the global goal that requires resolution. This maximizes the

	Overhead Ratio		
Phase	**Mean**	**Median**	**Max**
Encoding	0.979×	0.991×	1.120×
Solving	1.036×	1.011×	1.913×
Total	**0.998×**	**0.990×**	**1.699×**

Table 4: Performance overhead analysis (T_{wk}/T_{std}) on IPC problems. Ratios < 1.0 indicate the wk-encoding was faster.

	($\lvert\prec\rvert$)		Linearizations		
N	**Std**	wk	**Std**	wk	**Ratio**
3	18	16	66	90	1.36×
4	24	21	1,674	2,520	1.51×
5	30	26	6.9×10^4	1.1×10^5	1.63×
6	36	31	4.3×10^6	7.5×10^6	1.75×
7	42	36	3.7×10^8	6.8×10^8	1.85×
8	48	41	4.2×10^{10}	8.2×10^{10}	1.95×
9	54	46	6.1×10^{12}	1.3×10^{13}	2.05×
10	60	51	1.1×10^{15}	2.4×10^{15}	2.14×

Table 5: Comparison on the "Resource Borrowing" domain ($N = 3 \ldots 10$). "Ratio" denotes Lin_{wk}/Lin_{std}.

instantiation of the white knight disjunctions (Eq. 8).

Consequently, the encoding overhead was higher here than in the general evaluation: the wk-POCL encoding generated 24.0% more variables and 6.9% more clauses on average compared to the standard encoding. However, this increase in logical density did not translate to intractable solve times, as shown in Figure 3. The average solving time increased by only 11.6%, which in absolute terms corresponds to a negligible difference of approximately 12ms (0.112s vs 0.125s). This confirms that even in scenarios where the white knight machinery is heavily utilized, the optimization remains computationally efficient.

We note two important methodological details regarding these results. First, while counting linear extensions is generally #P-complete [Brightwell and Winkler, 1991], the structure of these instances allowed for exact enumeration within our timeout. Second, in general optimization scenarios, different partial orderings with the same number of constraints may yield different linearization counts due to solver tie-breaking. This was observed in the rovers, storage, and depot domains. However, that ambiguity does not apply here: the wk-POCL encoding achieved a *strict* reduction in the number of ordering constraints in the transitive closure in all cases (e.g., 51 vs 60). The results confirm that wk-POCL captures a significantly larger volume of the valid solution space as the causal complexity of the problem increases.

5 Conclusion

This paper re-examines the relationship between partial order (PO) plans and partial order causal link (POCL) plans, with a particular focus on the role played by the causal threat definition. Our main theoretical contribution is a proof that, under white knight semantics, the sets of valid PO plans and valid

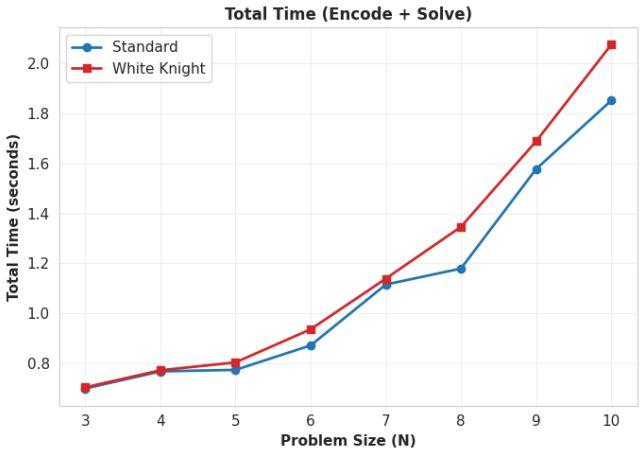

Figure 3: Time to solve Resource Borrowing domain.

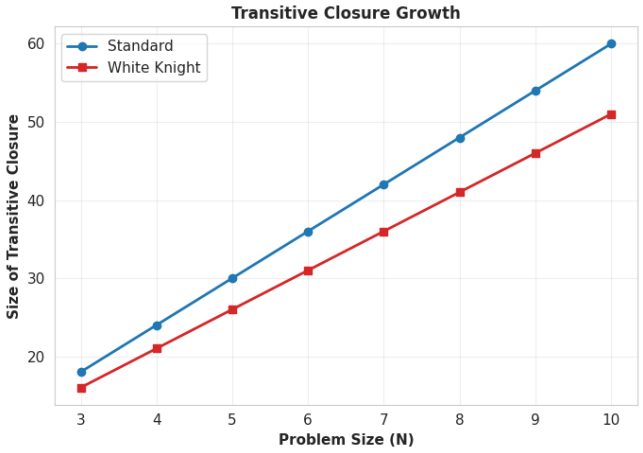

Figure 4: Number of ordering constraints in the transitive closure of the Resource Borrowing domain.

POCL plans coincide. Every PO plan admits a threat-free wk-POCL representation with identical ordering constraints, and every wk-POCL plan induces a valid PO plan. This result resolves the representational gap brought to renewed attention by Bercher and Olz [2020] and clarifies the conditions under which causal-link reasoning can be used without loss of generality for partial order planning.

This equivalence has direct implications for plan optimization. We show that MaxSAT-based approaches that claim to optimize PO plans (e.g., Muise *et al.* [2016]) in fact optimize over standard POCL plans due to their reliance on the classical threat definition. Adopting white knight semantics restores alignment between the optimization framework and the intended PO-plan solution space, with negligible computational overhead in practice.

In summary, we conclude that white knights are indeed worth the trouble: they reconcile the usefulness of POCL planning with the full generality of partial order plans, without sacrificing computational efficiency.

Acknowledgments

The authors thank Christian Muise for making POPGEN available and for helpful discussions. Pascal Bercher is the recipient of an Australian Research Council (ARC) Discovery Early Career Researcher Award (DECRA), project number DE240101245, funded by the Australian Government.

References

[Bercher and Olz, 2020] Pascal Bercher and Conny Olz. POP ≡ POCL, Right? Complexity Results for Partial Order (Causal Link) Makespan Minimization. In *AAAI 2020*, pages 9785–9793. AAAI Press, 2020.

[Bercher *et al.*, 2024] Pascal Bercher, Patrik Haslum, and Christian Muise. A Survey on Plan Optimization. In *IJCAI 2024*, pages 7941–7950. IJCAI Organization, 2024.

[Bercher, 2021] Pascal Bercher. A Closer Look at Causal Links: Complexity Results for Delete-Relaxation in Partial Order Causal Link (POCL) Planning. In *ICAPS 2021*, pages 36–45. AAAI Press, 2021.

[Bit-Monnot and Godet, 2025] Arthur Bit-Monnot and Roland Godet. Towards Canonical and Minimal Solutions in a Constraint-Based Plan-Space Planner. In *ECAI 2025*, pages 4678–4685. IOS Press, 2025.

[Bit-Monnot *et al.*, 2020] Arthur Bit-Monnot, Malik Ghallab, Félix Ingrand, and David E. Smith. FAPE: A Constraint-based Planner for Generative and Hierarchical Temporal Planning. arXiv preprint arXiv:2010.13121, 2020.

[Bit-Monnot, 2023] Arthur Bit-Monnot. Experimenting with Lifted Plan-Space Planning as Scheduling: Aries in the 2023 IPC. In *IPC 2023–HTN Tracks*, pages 7–9, 2023.

[Brightwell and Winkler, 1991] Graham Brightwell and Peter Winkler. Counting Linear Extensions Is #P-complete. In *STOC 1991*, pages 175–181. ACM Press, 1991.

[Bäckström, 1998] Christer Bäckström. Computational Aspects of Reordering Plans. *Journal of Artificial Intelligence Research*, 9:99–137, 1998.

[Chapman, 1987] David Chapman. Planning for Conjunctive Goals. *Artificial Intelligence*, 32(3):333–377, July 1987.

[Corrêa *et al.*, 2023] Augusto B. Corrêa, Guillem Frances, Markus Hecher, Davide Mario Longo, and Jendrik Seipp. Scorpion Maidu: Width Search in the Scorpion Planning System. In *IPC 2023–Classical Tracks*, 2023.

[Haslum *et al.*, 2019] Patrik Haslum, Nir Lipovetzky, Daniele Magazzeni, and Christian Muise. *An Introduction to the Planning Domain Definition Language*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Springer International Publishing, 2019.

[Ignatiev *et al.*, 2019] Alexey Ignatiev, Antonio Morgado, and Joao Marques-Silva. RC2: An Efficient MaxSAT Solver. *Journal on Satisfiability, Boolean Modelling and Computation*, 11(1):53–64, 2019.

[Kambhampati and Nau, 1996] Subbarao Kambhampati and Dana S. Nau. On the Nature and Role of Modal Truth Criteria in Planning. *Artificial Intelligence*, 82(1):129–155, 1996.

[McAllester and Rosenblitt, 1991] David McAllester and David Rosenblitt. Systematic Nonlinear Planning. In *AAAI 1991*, pages 634–639. AAAI Press, 1991.

[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.

[Muise *et al.*, 2022] Christian Muise, Florian Pommerening, Jendrik Seipp, and Michael Katz. Planutils: Bringing Planning to the Masses. In *ICAPS 2022 System Demonstrations*, 2022.

[Oates and Bercher, 2026a] Harrison Oates and Pascal Bercher. Experimental Setup for the IJCAI-ECAI 2026 Paper: "Are White Knights Worth the Trouble? Reconciling POCL and Partial-Order Plans for Plan Optimization". doi: 10.5281/zenodo.1995010, 2026.

[Oates and Bercher, 2026b] Harrison Oates and Pascal Bercher. Makespan Investigations of Sequential, Parallel, PO, and POCL Plans. In *AAAI 2026*, pages 36334–36342. AAAI Press, 2026.

[Oates and Bercher, 2026c] Harrison Oates and Pascal Bercher. Relaxing is Hard: Complexity Results for Lifted Partial Order Causal Link Planning. In *ICAPS 2026*. AAAI Press, 2026.

[Penberthy and Weld, 1992] J. Scott Penberthy and Daniel S. Weld. UCPOP: A Sound, Complete, Partial Order Planner for ADL. In *KR 1992*, pages 103–114. Morgan Kaufmann Publishers Inc., 1992.

[Siddiqui and Haslum, 2015] Fazlul Hasan Siddiqui and Patrik Haslum. Continuing Plan Quality Optimisation. *Journal of Artificial Intelligence Research*, 54:369–435, 2015.

[Wilcoxon, 1945] Frank Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, 1945.