Reviving Partial Order Causal Link (POCL) Planning: Is It Possible? Is It Worth It?

Scott Howsam, Harrison Oates, Pascal Bercher

School of Computing, The Australian National University, Canberra
u7115241@alumni.anu.edu.au, harrison.oates@anu.edu.au, pascal.bercher@anu.edu.au

Abstract

Partial Order Causal Link (POCL) planning has been largely abandoned for classical plan generation tasks, as state-based methods consistently outperform POCL planners in most benchmark domains. However, POCL planning offers theoretical advantages—compact plan representations, least-commitment search, and explicit causal structure—that have motivated several revival attempts over the past three decades. This position paper provides a systematic analysis of why we believe these attempts, which primarily employed delete-relaxation heuristics, failed to achieve competitive performance. We identify two fundamental challenges: first, computing informed heuristics that respect POCL search node structure is NP-hard even for delete-relaxation approaches; second, POCL search nodes grow quadratically with search depth, compounding heuristic computation costs despite progress towards the goal. We then analyze how these challenges might extend to unexplored heuristic families, including landmark-based, pattern database, and width-based approaches, identifying this as an open question warranting future investigation.

Introduction

Partial Order Causal Link (POCL) planning uses partial plans as search nodes to maintain a principle of least commitment, postponing action ordering and parameter binding decisions until required (Weld 1994). Causal links explicitly represent causal relationships between actions, providing rich structural information that can guide search and enable pruning. These properties made POCL planning the dominant paradigm through the early 1990s, and continue to make it valuable in contexts including plan optimisation (Bercher, Haslum, and Muise 2024; Oates and Bercher 2026b); HTN and hybrid planning (Bercher, Lin, and Alford 2022); and temporal planning, where approaches range from forward-chaining planners that incorporate partial-order reasoning (Coles et al. 2010) to constraint-based planners that maintain full POCL structures (Bit-Monnot et al. 2020; Bit-Monnot and Godet 2025).

However, original research into POCL planning focused on using purely POCL-based planners for classical plan generation. POCL planners offer structural advantages that theoretically reduce the size of the search space compared to state-based planners. The least-commitment principle prevents premature commitment to action orderings and variable instantiations, potentially avoiding extensive exploration of dead-end paths. A single partial plan can represent up to a factorial number of total-order action sequences, allowing POCL search trees to be exponentially smaller than their state-based counterparts in favorable cases (Minton, Bresina, and Drummond 1994). Causal links provide additional pruning power by enforcing adherence to established causal relationships (Bercher 2021). Despite these advantages, POCL planners began to underperform state-based and SAT-based planners as heuristic search methods matured, and by the late 1990s POCL planning was largely abandoned for classical plan generation tasks (Weld 1999).

We provide a systematic analysis of the structural challenges underlying this underperformance, focusing on heuristics: unlike pruning strategies or flaw-selection policies, heuristics determine how effectively search is guided towards the goal, and it is precisely this guidance that state-based planners excelled at and POCL planners consistently failed to match. Crucially, we argue that identifying why previous revival attempts failed is itself a constructive contribution: obstacles that are precisely characterised can be deliberately targeted, and future approaches that address them directly have a better chance of making POCL planning competitive than those that do not. We identify two core issues. First, computing delete-relaxation heuristics that respect POCL search node structure is NP-hard (Bercher 2021), whereas the same heuristics are polynomial-time for state-based planning. Second, POCL search node size grows quadratically with plan length, while state-based node size remains constant. To our knowledge, no prior work has connected these two properties to explain why conceptually different POCL heuristics failed in similar ways. For delete-relaxation heuristics they are fundamentally intertwined: escaping the first requires exploiting node structure, but exploiting node structure means confronting the second. This persistent tradeoff explains the consistent failure of delete-relaxation-based revival attempts, and — by identifying precisely what each future direction must overcome — points toward where a successful revival might begin. We classify POCL heuristic approaches by how they navigate this tradeoff, revealing that all delete-relaxation approaches fall into predictable categories, each facing a foreseeable obstacle. This also highlights unexplored families — landmark-based, pattern database, width-based, and incremental methods —

Direction	Obstacle
Incremental computation	Node growth
Width-based heuristics	Tractability
Pattern databases	Both
Operator counting	Both
Dusting off classics	—
Hybrid/Bounding Plan Size	Node growth
Total-order plan-space	Tractability

Table 1: Proposed future directions and the obstacles each targets. *Tractability*: informed heuristics are NP-hard to compute (Thm. 1). *Node growth*: search node size grows quadratically with depth (Thm. 2).

and for each we analyze what the tradeoff implies about its prospects. Table 1 summarises each proposed future direction alongisde the obstacle it primarily targets.

Formal Framework

We consider propositional STRIPS-style planning (Fikes and Nilsson 1971), loosely following the POCL framework of Bercher and Olz (2020). A *classical planning problem* $\mathcal{P}$ is a tuple (V, A, s_I, g), where V is a finite set of propositional state variables, A is a finite set of *actions*, $s_I \in 2^V$ is the initial state, and $g \subseteq V$ is the goal description. Each action $a \in A$ has a *precondition* list $pre(a)$, *add* list $add(a)$, and *delete* list $del(a)$, all subsets of V. Action a is executable in state $s \in 2^V$ if $pre(a) \subseteq s$, producing state $\gamma(a, s) = (s \setminus del(a)) \cup add(a)$. A *total-order plan* for $\mathcal{P}$ is an action sequence executable in s_I whose result is a goal state (i.e., $s \supseteq g$).

A *partial POCL plan* $P = (PS, \prec, CL)$ consists of a finite set of *plan steps* PS, each pairing a unique label with an action, a *partial order* $\prec$ on PS, and a finite set of *causal links* CL. As with actions, we use $pre(ps)$, $add(ps)$, and $del(ps)$ to refer to the precondition, add, and delete lists of a plan step's action. A causal link $(ps_p, v, ps_c) \in CL$ records that ps_p *produces* variable v for *consumer* ps_c, where $v \in add(ps_p) \cap pre(ps_c)$. The link *protects* v: no step that deletes v may be ordered between ps_p and ps_c in any valid linearization. A precondition $v \in pre(ps_c)$ is *open* if no causal link (ps_p, v, ps_c) exists in CL.

Causal links serve two roles. First, the set of open preconditions identifies what remains to be achieved. Second, causal links provide *pruning power* by constraining valid linearizations: a plan step ps_t *threatens* a link (ps_p, v, ps_c) if $v \in del(ps_t)$ and ps_t could be ordered between ps_p and ps_c. Such threats are *flaws* that must be resolved by ordering ps_t before ps_p or after ps_c. For example, Figure 1, without the ordering constraint the action A3 threatens the link between A1 and A2. It is this pruning power that makes reasoning about POCL plans computationally challenging (Bercher 2021). Each causal link induces an ordering constraint $ps_p \prec ps_c$.

The problem's initial state and goal are encoded as actions $init$ and $goal$ with $add(init) = s_I$ and $pre(goal) = g$ (all other effect and precondition sets empty), ordered so that

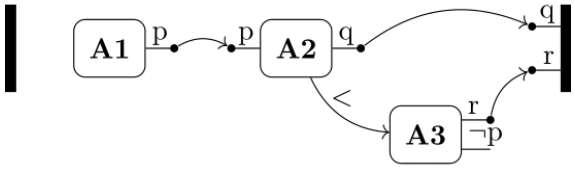

Figure 1: An example of a partial-order plan where orderings have resolved threats.

$init$ precedes and $goal$ follows all other steps.

Definition 1. A partial POCL plan P is a *POCL solution* for a classical planning problem iff every precondition is supported by a causal link and there are no causal threats.

We note that partial-order and POCL plans are used interchangeably in the literature, though strictly speaking they are not equivalent under the standard threat semantics. The two can be made equivalent, however (Oates and Bercher 2026a), and this distinction does not affect our analysis.

To define heuristics over POCL search nodes, we need to capture partial-plan structure. A *POCL planning problem* $\mathcal{P} = (V, A, P_{s_I,g})$ augments the variable and action sets with an initial partial plan $P_{s_I,g} = (PS_I, \prec_I, CL_I)$ containing the artificial steps $\{init, goal\} \subseteq PS_I$. Each search node corresponds to such a problem, with $P_{s_I,g}$ representing the partial plan constructed so far. A POCL solution to $\mathcal{P}$ must be a *refinement* of $P_{s_I,g}$—adding steps, orderings, and causal links without modifying existing ones—that satisfies Definition 1. This framework of repeated refinements was introduced by Kambhampati, Knoblock, and Yang (1995); Kambhampati (1997) as *refinement planning*.

Previous Revival Attempts

Compact representations and the principle of least commitment were used as justifications for the superiority of partial-order planning through the early 1990s (Weld 2011), which led to a research focus on partial-order planners. Minton, Bresina, and Drummond (1994) challenged this conventional wisdom, noting that "the superiority of partial-order planning can depend critically upon the search strategy and search heuristics employed." Developments in state-based planning allowed for highly informed and efficiently computable heuristics, whereas partial-order planning struggled to produce comparable guidance. This insight catalyzed a shift in planning research away from POCL, beginning with Graphplan (Blum and Furst 1995), which used parallel plans rather than PO or POCL plans, and forward state-based planners such as UNPOP (McDermott 1996) which began outperforming partial-order planners. By 1999, Weld (1999) observed that POCL planners were receiving diminished attention due to being outperformed by other methods, and it is from here that research into partial-order planning begins to be seen as revival attempts.

A major reason for the success of early state-based planners such as UNPOP (McDermott 1996) and later the FF planning system (Hoffmann and Nebel 2001) were their well informed, efficiently computable heuristics. POCL revival

attempts accordingly focused on creating similarly fast and informed heuristics for POCL search. These attempts can be organized into two broad categories based on how they navigate the fundamental tension between heuristic computation cost and heuristic informativeness. We first summarize the key approaches that have been tried, proceed to a systematic analysis of the limitations they faced, and finally assess whether alternative ideas may offer a viable path forward.

Heuristics Using Limited Node Information

Early POCL revival attempts focused on creating heuristics that were as informed as possible while remaining polynomial-time and avoiding significant scaling factors as node size increases. This approach solves the node size scaling issue we will discuss in the next section, but does so by ignoring or relaxing information contained within the current search node.

Delete-relaxation heuristics Nguyen and Kambhampati (2001) present an early POCL revival attempt with their RePOP (Revived Partial Order Planning) planner, which introduced the Relax heuristic. This heuristic adapts the relaxed-plan heuristic from FF (Hoffmann and Nebel 2001) to partial plans. It constructs a planning graph and extracts a relaxed plan by ignoring delete effects, recursively expanding backward from the goal. At each step, it inserts the delete-relaxed action that adds the highest-level proposition remaining in the goal state, then updates to a new goal state. Eventually this process reduces to only the costs of actions and propositions in the initial state, at which point the heuristic value is calculated as the sum of all added actions that were not already in the partial plan. This heuristic is very fast to compute and outperformed previous partial-order heuristics in most domains, but did not outperform AltAlt (Nguyen, Kambhampati, and Nigenda 2002), the state-based planner against which it was evaluated.

FF (Hoffmann and Nebel 2001) estimates goal distance by ignoring delete lists in state-based planning. However, since previous deletions are already encoded within the current state, this relaxation implicitly only applies to *future* actions, not relaxing the current search node at all. In contrast, the Relax heuristic ignores all partial orderings, the pruning power of causal links, and delete effects in the *current search node*, which represents a significant loss of information.

The additive heuristic h_{add} (Bonet, Loerincs, and Geffner 1997; Bonet and Geffner 2001) was similarly adapted to POCL planning (Younes and Simmons 2002) and deployed in the VHPOP planner (Younes and Simmons 2003). This heuristic takes a different approach from Relax: rather than extracting a relaxed plan at search time, it precomputes a cost for each ground literal during preprocessing and then, for each partial plan, sums the precomputed costs of all open conditions. This makes it computationally efficient but introduces the assumption of subgoal independence—each open condition is costed independently, so the heuristic cannot recognize when multiple open conditions might be satisfied by the same future action. The version deployed in VHPOP, denoted h^r_{add}, adds a check for reuse of existing actions: for each open condition, the heuristic examines whether any action already in the partial plan could satisfy it, using ordering constraints to verify that the candidate provider is not constrained to execute after the consumer. This systematic use of ordering information contrasts with the Relax heuristic, which relaxes ordering constraints and only accounts for existing actions opportunistically during plan extraction. The two heuristics thus represent different tradeoffs. Relax can recognize positive interactions among potential future actions (reducing overestimation when multiple goals share achievers) but handles existing actions less systematically. The Additive heuristic treats future actions independently (potentially overestimating when goals share achievers) but more thoroughly exploits opportunities to reuse existing actions. Empirically, h^r_{add} significantly outperformed Relax in domains where systematic reuse checking dominates (such as gripper), while Relax performed slightly better in domains where shared achievers among future actions are common. Neither heuristic exploits the pruning power of causal links, and both ignore delete effects in their relaxation.

While both RePOP and VHPOP showed improvement over previous POCL planners, and over older total-order planners, they were outperformed by state-based and SAT-based planners published around the same time, such as the FF planning system (Hoffmann and Nebel 2001). It is notable that pre-existing causal links and delete effects both contain significant pruning power and information relevant to the direction of search, and yet both heuristics ignore them entirely. This valuable information takes time for the search process to find, and the information it holds is a large reason why partial order planning was previously seen as superior. The equivalent in state-based planning would be discarding or relaxing parts of the current state, which would likely be seen as ridiculous (Bercher 2021). These heuristics essentially apply state-based heuristic techniques to POCL planning without exploiting the additional information contained within POCL search nodes, treating POCL nodes as state-based nodes with extra unused structure.

Sampling-Based Approaches The Sample-FF heuristic (Bercher et al. 2013) adopts a different strategy for achieving polynomial-time computation. Rather than relaxing structural information from the partial plan, it addresses the NP-hardness of delete-relaxed action insertion over partial orders through sampling: it uses Markov Chain Monte Carlo to draw a constant number of total-order linearizations consistent with the current ordering constraints, computes a delete-relaxation estimate for each by constructing relaxed planning graphs between consecutive steps (as done with state-based FF), and returns the minimum of the heuristic values.

A variant exists to respect causal links by filtering out actions that would threaten them during relaxed planning graph construction. However, the empirical results reveal that when respecting all causal links, 36-46% of search nodes produced *no* feasible relaxed solution from *any* sampled linearization, compared to only 1-4% when causal links were ignored entirely. This difference explains why the configuration without causal link filtering solved significantly more problems (187 of 446) despite being theoretically less informed. The heuristic thus does not exploit the pruning

power of causal links, even though it has access to them; indeed, adding more structural information makes it worse.

The fundamental limitation of sampling emerges from the gap between sampled and actual linearization spaces. The ordering constraints and causal structure of partial plans heavily restrict which linearizations are feasible even under delete relaxation. Critically, as a partial plan becomes more informed, the feasible space shrinks. This is exactly when a POCL heuristic should become *more* accurate. Instead, Sample-FF becomes more likely to miss all feasible linearizations entirely. When all sampled linearizations happen to be infeasible, the heuristic falls back to simply counting open preconditions, becoming largely uninformative. This can occur even on branches leading to optimal solutions, since feasible linearizations may exist but fail to be sampled. Thus, the more information the search accumulates, the more brittle and blind the heuristic becomes.

Sample-FF was shown to be competitive with the Additive and Relax heuristics in some domains but could not consistently outperform either, and did not approach the performance of state-based planners. The sampling approach thus trades structural relaxation for incomplete coverage of the linearization space, and encounters similar limitations through this different path, while adding the further drawback that more informed partial plans systematically undermine its effectiveness rather than improving it.

Heuristics Using Most Node Information

Some POCL revival attempts aimed to avoid significantly relaxing pre-existing search node information, creating more informed POCL heuristics that either did not relax current search node information or ensured the relaxation did not ignore entire POCL search node attributes such as delete effects or causal links. This helped to maintain the pruning power provided by these attributes, but at the cost of increased computational complexity.

Linear Programming: The Lplan Heuristic The first example of an admissible POCL heuristic was the Lplan heuristic (Bylander 1997), which exploits the fact that propositional planning problems are NP-complete if plans are limited to a polynomial number of steps (Turner 2002). It formulates an integer linear program (ILP) that can find a solution plan of predefined length, outputting "infeasible" if no solution exists at that length. By iteratively checking each possible solution length starting from zero, this method can solve any feasible planning problem. For use as a heuristic, the ILP is relaxed to a linear program (LP).

As an LP relaxes the requirement for integer variable values, this allows for fractional solutions, such as half of two different actions being executed during one time step. Such a linear relaxation is solvable in polynomial time (Khachiyan 1979). However, for a polynomial-time problem, it is often very difficult to solve, as evidenced by debate over whether the problem was in P or NP only a year before Khachiyan's algorithm was published (Lawler and Labetoulle 1978). Since the Lplan heuristic requires an LP to be solved for each plan length until a solution is found, this computational cost increases further. This leads to the Lplan heuristic not being competitive with the state of the art at the time.

However, Lplan does not ignore any POCL search node information except causal links. The power of this information in reducing the search space is shown in the results, as the number of explored nodes to find a solution was orders of magnitude smaller than UCPOP (Penberthy and Weld 1992) for some domains, even though total computation time was higher due to the LP solving overhead. We observe that causal links can in fact be incorporated into Lplan through a polynomial-time compilation into additional state variables.

Proposition 1. *Given a POCL planning problem $\mathcal{P} = (V, A, P_{s_I,g})$ with causal links CL_I, causal link protection can be encoded into state-based planning in polynomial time. For each plan step (ps, a), a fresh action a_{ps} is introduced with the same preconditions and effects as a. For each causal link $cl = (ps_p, v, ps_c) \in CL_I$, a fresh variable v_{cl} is introduced such that $v_{cl} \in add(init)$, $v_{cl} \in del(a_{ps_p})$, $v_{cl} \in add(a_{ps_c})$, and for all actions a' in the augmented action set A': $v \in del(a') \Rightarrow v_{cl} \in pre(a')$.*

In the compilation, the variable v_{cl} acts as a guard: it is initially true, becomes false when the producer ps_p executes, and is restored when the consumer ps_c executes. Any action that deletes the protected literal v requires v_{cl}, so it cannot execute during the interval when v_{cl} is false—exactly the interval between producer and consumer. The compilation introduces $|CL_I|$ variables and adds at most one precondition per causal link to each action that deletes the protected literal, for a total overhead of $O(|CL_I| \cdot |A|)$.

State-Based Heuristic Conversion Another approach to utilizing more search node information is the state-based heuristic conversion method by Bercher, Geier, and Biundo (2013). This encodes a partial-plan search node as a classical planning problem, then calculates a state-based heuristic on the initial state of this problem. The authors describe a compilation for causal links using counter variables, but this encoding is exponential in the maximal number of preconditions and effects per action, and was not included in their empirical evaluation. The polynomial-time causal link compilation we have described could be substituted, providing the same information more efficiently.

Although the encoding translates *all* POCL information into a classical state-based problem without information loss, the deployed classical heuristics must relax some of this plan information. Depending on which state-based heuristic is used, this approach could relax original plan delete effects, causal links, orderings, potential linearizations of the partial plan, or some combination of the above. LM-Cut, a landmark-based heuristic (Helmert and Domshlak 2009), and *Merge and Shrink*, an abstraction based heuristic (Helmert, Haslum, and Hoffmann 2007), were tested on the encoded problem (without causal links) by Bercher, Geier, and Biundo (2013). Due to implementation overhead from calling an external planner for each heuristic evaluation, runtime was dominated by this overhead rather than search. However, measured by search space size, LM-Cut was more informed than the Additive heuristic across all problems it solved. The authors note that a native implementation of LM-Cut for POCL planning could offer sig-

nificant performance improvements, though this has not yet been pursued as far as we are aware.

Summary of Historical Approaches

The revival attempts from 1997 to 2013 explored various points along the tradeoff between heuristic informativeness and computational cost. Polynomial-time approaches like Relax and Additive achieved tractability by relaxing structural information from partial plans—ignoring delete effects, causal link pruning power, or subgoal interactions—but could not match state-based planner performance despite significant improvements over earlier POCL heuristics. Lplan preserved more information and achieved orders-of-magnitude reductions in node expansions, but LP solving overhead at each node made it uncompetitive in wall-clock time. Sample-FF attempted to preserve some search node structure through sampling, but incomplete coverage of the linearization space introduced its own limitations.

However, these results do not foreclose the possibility of competitive POCL planning. The state-based heuristic conversion experiments by Bercher, Geier, and Biundo (2013) offer a particularly suggestive result: LM-Cut, a landmark-based heuristic, produced smaller search spaces than the Additive heuristic across all problems it solved. The approach was not competitive in runtime largely due to implementation overhead from calling an external planner at each node—overhead that a native implementation would eliminate. The authors identified this as a promising direction, yet no such implementation has been pursued in the subsequent decade. Likewise, advances in LP technology substantially weaken the historical argument against richer POCL heuristics: modern LP solvers are roughly $9\times$ faster algorithmically than their 2001 counterparts and, when combined with hardware improvements, deliver around a $180\times$ speed-up overall (Koch et al. 2022). These gains suggest that previously uncompetitive approaches may now be entirely viable.

More broadly, the revival attempts focused almost exclusively on delete-relaxation heuristics. Other heuristic families that have proven effective in state-based planning—including landmarks (Helmert and Domshlak 2009), pattern databases (Haslum et al. 2007), operator counting (Pommerening et al. 2014), and width-based approaches (Lipovetzky and Geffner 2012)—remain largely unexplored for POCL planning. Whether these families can escape the tradeoffs that constrained delete-relaxation approaches is an open question. To make progress on it, we first need to understand *what* these tradeoffs are and *why* they arise as structural consequences of POCL planning. In the following section, we identify two challenges that together explain the consistent pattern of failure across conceptually different heuristic approaches, and with which any future direction must reckon.

Theoretical Challenges in POCL Planning

Building upon our analysis of previous revival attempts, this section formalizes two fundamental challenges that have contributed to POCL revival attempts not succeeding in classical settings so far: the difficulty of creating informed heuristics that respect POCL search node structure, and the quadratic growth of search node size with depth. We show that these challenges are not merely implementation difficulties but arise from the structural properties of POCL planning itself, and that they are fundamentally intertwined in a way that creates a seemingly inescapable tradeoff.

The Heuristic Information Challenge

The first major challenge is the computational complexity of fully exploiting POCL search node information. In state-based planning, heuristics relax the *problem* but not the current *search node*—the state itself is used as-is. A POCL node contains substantially more information than states, but exploiting this power increases computational complexity. Bercher (2021) provided a comprehensive complexity analysis of delete-relaxed POCL planning, establishing tight bounds for various relaxations of the POCL framework. They find two tractable cases.

Theorem 1 (Polynomial Cases of Delete-Relaxed POCL (Bercher 2021))**.** *POCL planning with delete-relaxed insertable actions is solvable in polynomial time if* either*:* $\prec$ *is a total ordering (making plans sequential),* or *The initial partial plan is delete-relaxed and initial causal links are not respected. Otherwise, it is NP-complete.*

These conditions characterize exactly when the POCL heuristics previously described can be computed efficiently: the first is exploited by Sample-FF, the second by RePOP's Relax and VHPOP's Additive heuristics. The implication of this result is that any known polynomial-time POCL heuristic must either abandon the partial-order structure or ignore the causal and delete-effect information that makes POCL planning theoretically attractive.

The Node Growth Challenge

The second major challenge for POCL planning is that search node size grows with search depth. In state-based planning, each node in the search tree is a state $s \in 2^F$, whose size $|s| \leq |F|$ is bounded by the number of fluents in the problem. As search progresses deeper into the tree, states may change, but they do not grow in size – a state at depth 100 has the same representation size as a state at depth 1.

POCL planning operates differently. A partial plan $(PS, \prec, CL)$ is built incrementally by adding plan steps, ordering constraints, and causal links. Since POCL search never removes information—only adds refinements—partial plans can only grow in size as search depth increases. Thus search depth and node size are tightly coupled. Bäckström and Jonsson (2011) compare the complexity of partial-order planning to total-order planning, and conjecture that partial-order planning is only harder than total-order planning because of this growth in search node size. This growth is trivial in degenerate cases. Any total-order planner can be reinterpreted as a partial-order planner, and in such cases each refinement adds exactly one step and one ordering constraint, yielding linear node growth.

To reason about partial-order planning in a way which excludes such cases, Bäckström (1998) defined a set of *optimality criteria* that any genuine least-commitment planner

must satisfy. These criteria forbid adding orderings that are not justified by causal threats or causal structure. They ensure that the partial order remains minimal. Under these criteria, node growth cannot remain linear.

Theorem 2 (Quadratic growth of ordering constraints)**.** *There exist planning instances with n steps such that any partial plan satisfying at least one of Bäckström's criteria must contain $\Omega(n^2)$ ordering constraints. Hence, in the worst case, partial-plan representations sizes grow quadratically with search depth.*

Proof sketch. Bäckström's criteria forbid unnecessary orderings; thus the planner must leave two steps unordered unless doing so violates causal or threat-resolution requirements. However, there exist instances in which threats to causal links force an ordering between every pair of plan steps. For example, consider a chain of actions where each a_i requires p_{i-1}, adds p_i, and deletes p_j for all $j \in \{1, \ldots, n\} \setminus \{i-1, i\}$. To protect earlier actions' preconditions from later actions' deletions, all $\binom{n}{2}$ pairs must be ordered. Since Bäckström's criteria only allow necessary orderings, and all these orderings are necessary, any minimal partial order must contain $\Omega(n^2)$ constraints. □

While this is a worst-case theoretical result, the behaviour appears empirically across a range of domains. To test this, we conducted two experiments on an AMD Ryzen 7 7800x3D CPU with 4.2GHz base clock. In the first, we instrumented VHPOP to record plan statistics at every generated search node and ran the default configuration with a one minute timeout on the first ten instances of *Blocks*, *Gripper*, *Logistics*, and *Satellite*. For each partial plan encountered during search, we recorded the number of actions and ordering constraints, and report the mean number of ordering constraints as a function of plan size (number of actions), averaged over all search nodes and then across problems within each domain. The domains differ substantially in causal density, interaction patterns, and threat structure. Despite these structural differences, all domains exhibit growth well-described by a quadratic model, with $R^2 \geq 0.98$ across all four domains (Figure 2). In our second experiment, we instrumented VHPOP to record the heuristic evaluation time at every node across the same domains. Using h^r_{add}, valuation time similarly follows a quadratic trend in three of four domains; the exception is *Satellite* ($R^2 = 0.24$), likely reflecting its lower causal density reducing the coupling between plan length and evaluation time (Figure 3). Fit statistics for both experiments are reported in Table 2.

Planner design choices such as flaw selection and threat resolution strategies influence which refinements are explored (Younes and Simmons 2003) and therefore affect constant factors in node growth. However, they do not alter the underlying source of coupling: when many steps interact via causal threats, extending a partial plan necessarily introduces additional ordering constraints. Similarly, while lifted representations may defer variable binding and may reduce the number of plan steps slightly (Oates and Bercher 2026c), the ordering constraints arising from causal threats still accumulate over the same underlying causal structure.

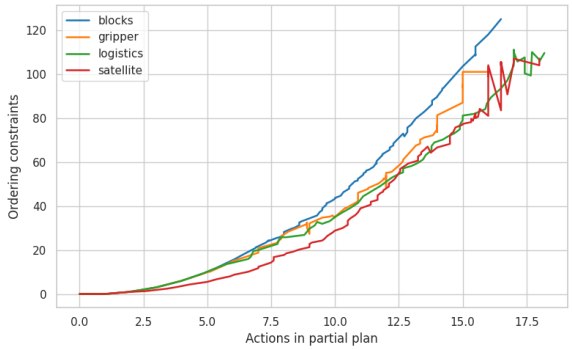

Figure 2: Growth of ordering constraints with partial plan size (mean across problems per domain).

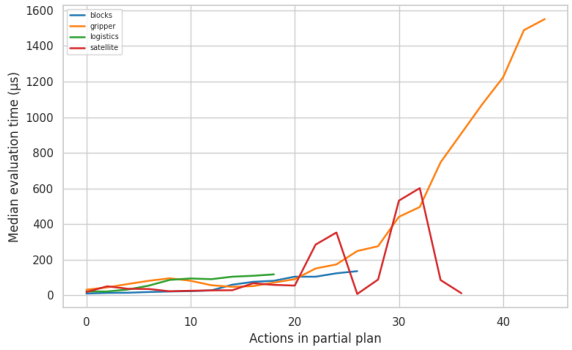

Figure 3: Growth of median evaluation time for $h^r_{add}$with partial plan size (median across problems per domain)

Even for polynomial-time heuristics, this growth in node size has a direct cost: as seen in Figure 3, heuristic evaluation time increases steadily with plan size, and the structural information that is being ignored grows alongside it, wasting an increasing amount of the causal and ordering detail that makes POCL planning attractive. For heuristics that do exploit this information, the situation is further compounded by Theorem 1: such heuristics must solve NP-hard problems on inputs of size $\Omega(n^2)$, meaning that the closer the search comes to a goal, the harder each heuristic evaluation becomes, which is clearly a major disadvantage.

The Informativeness-Tractability Tradeoff

Theorems 1 and 2 together reveal why POCL planning faces systematic obstacles in classical domains: *the only way to achieve tractable heuristics is to abandon the information that makes POCL planning attractive, but that information compounds in cost quadratically as search progresses.*

More precisely: informed POCL heuristics require NP-hard computation (Theorem 1), while least-commitment orderings require $\Omega(n^2)$ constraints for n actions (Theorem 2). Thus informed heuristics operate on inputs growing quadratically with depth, making reasoning about goal distance *harder* the closer we come to the goal – a paradoxical and counter-intuitive situation for a search algorithm. For heuristics computed from scratch at each node, the only es-

Domain	R^2 (Fig. 3)	R^2 (Fig. 4)
Blocks	0.9992	0.9717
Gripper	0.9924	0.9799
Logistics	0.9923	0.9486
Satellite	0.9820	0.2371

Table 2: Coefficients of determination (R^2) for quadratic fits to ordering constraint and heuristic evaluation time growth.

cape is polynomial-time heuristics that relax orderings and causal links, but these are exactly the structures distinguishing POCL from state-based planning. This partly explains the systematic pattern in revival attempts: relaxation-based approaches (RePOP, VHPOP, Sample-FF) sacrifice informativeness for tractability, while informed approaches (Lplan) did not scale despite dramatically reducing node expansions.

Potential Directions for POCL Planning

The challenges identified above apply specifically to delete-relaxation heuristics computed from scratch at each node. Two observations suggest alternatives: heuristics outside the delete-relaxation family are not subject to Theorem 1's NP-hardness result, and each individual refinement adds only a small increment to the partial plan, creating opportunities for incremental computation that avoids confronting the full node size. We outline several unexplored directions.

Incremental Computation The most direct response to the node growth challenge is to avoid recomputation. The structure of refinement planning is reminiscent of incremental SAT solving, where learned clauses are maintained across related problems rather than discarded after each query. An incremental approach to POCL heuristics would maintain heuristic information across refinements, updating it to reflect the small change rather than recomputing on the full $\Omega(n^2)$ structure. By updating only the part of the heuristic affected by the refinement, the amortized cost per node could depend on the refinement size rather than the total node size, reducing practical overhead even if the theoretical complexity remains unchanged. This approach is compatible with best-first search: heuristic values are computed when a node is generated as a child of the currently expanded node, so the incremental update is always from parent to child—a single refinement step—regardless of expansion order. Each open-list node must store the heuristic state needed to seed its children's evaluations, but since POCL search already maintains the full partial plan at every node, the additional overhead may be modest in comparison.

No systematic investigation of incremental heuristic computation for POCL planning exists, despite incremental approaches proving effective in state-based planning (Pommerening and Helmert 2012). This is a significant gap, given the similarities to refinement planning.

Width-based approaches Width-based planning, exemplified by $IW(i)$ and best-first width search (Lipovetzky and Geffner 2012), measure the 'novelty' of a state (whether it achieves atom tuples not seen earlier in the search) rather than how far a state is from the goal. States with high novelty are prioritized, leading to effective exploration without explicit goal-distance computation.

A novelty measure for partial plans would ask whether it contains structures not seen in previously explored plans. The richer structure of partial plans offers multiple dimensions for this: flaw novelty (open conditions or threats involving new literals), causal link novelty (novel producer-consumer-literal tuples), and ordering novelty (new ordering constraints between action types). Tracking these structures is a different computational challenge than goal-distance estimation, and one that scales with the novelty threshold rather than node size. Width-based techniques have shown surprising effectiveness in state-based planning, often outperforming delete-relaxation heuristics on specific problem classes, yet their adaptation has not been attempted.

Pattern Database and Abstraction Heuristics Pattern database heuristics precompute goal distances for abstract versions of the problem, then use table lookups during search. In state-based planning, this yields constant-time heuristic evaluation after preprocessing. The Merge-and-Shrink framework generalizes this approach, constructing abstractions that can be more expressive than standard pattern databases (Sievers and Helmert 2021). The state-based heuristic conversion by Bercher, Geier, and Biundo (2013) applied Merge-and-Shrink through the encoding, but this approach is indirect: the abstraction is computed over the encoded classical problem rather than the POCL structure itself. A native abstraction for partial plans might abstract over the space of partial orders or causal structures directly, potentially capturing regularities that the encoding-based approach misses. What such abstractions would look like, and whether they could achieve the constant-time lookup that makes pattern databases attractive, remains unexplored.

Operator Counting Operator counting is a heuristic framework developed for state-based planning that expresses goal-distance estimation as a linear programming problem (Pommerening et al. 2014). For each action in the planning problem, a count variable represents how many times that action appears in a plan. The LP objective minimizes total action count (or cost), while constraints encode requirements that any valid plan must satisfy. Different heuristic information translates into different constraints on these count variables. Landmark constraints require that if a fact must be true at some point in any valid plan, then at least one action achieving that fact must appear. Flow constraints enforce that the production and consumption of each fact must balance across the plan. Delete-relaxation bounds and abstraction-derived information can similarly be encoded.

The power of this approach lies in unification: constraints from different heuristic families can be combined in a single LP, producing bounds tighter than any individual source. For example, landmarks and abstractions capture complementary information, and their combination achieves synergies unavailable to either alone. Operator counting has not been explored for POCL planning. However, Lplan demonstrates that linear programming is a viable foundation for POCL heuristics, and operator counting can be seen as a more gen-

eral framework that could extend this foundation.

Dusting Off the Classics As discussed in our analysis of previous revival attempts, several approaches showed promise but may have been hampered by implementation. We believe that these approaches merit reinvestigation with modern infrastructure.

Compromises Worth Considering

The directions outlined above represent potential paths toward making pure POCL planning competitive. Whether any of these can overcome the challenges identified in this paper remains to be seen. If not, a pragmatic alternative is to partially retreat, and to ask which of POCL's commitments are essential and which can be strategically abandoned. The following directions sacrifice either the partial-ordering or the full plan-space representation, but preserve aspects of POCL structure that state-based planning lacks entirely.

Hybrid Approaches A straightforward response to the node growth challenge is to bound the size of any POCL plan ever produced during search. Once nodes are prevented from growing beyond a fixed threshold, heuristic evaluation costs are similarly capped. There are many mechanisms by which this could be achieved. One option is to transition out of POCL search once a node exceeds the threshold, encoding the partial plan as a classical planning problem using the compilation of Bercher, Geier, and Biundo (2013) – extended with our polynomial link encoding – and solving via state-based search from that point forward. Another option might be to spawn a fresh POCL subproblem from the current partial plan to make sub-plans which are subsequently stitched together. A third option is to compress the existing plan by fusing together pairs of actions (say, a and b with $a \prec b$) into a single composite step, merging their preconditions, effects and causal links, thereby reducing node size while preserving all the essential relationships. These are illustrative possibilities; the key point is that the abstract objective of preventing unbounded node growth is robust even if any particular realization proves difficult. We believe this objective deserves attention as a design principle in its own right, independent of which mechanism is ultimately used to enforce it. Such hybrid approaches have not been systematically explored in the literature.

Total-Order Plan-Space Planning A planner could bypass a source of hardness in Theorem 1 by abandoning the principle of least-commitment for action orderings, but retaining flexibility to insert actions anywhere in the current plan sequence. Such a planner would maintain causal links and take advantage of their pruning power while committing to a total ordering of actions rather than a partial order. This represents the opposite trade-off to forward-chaining PO planning approaches, which perform state-based search while deferring ordering commitments wherever temporal constraints allow (Coles et al. 2010). To our knowledge, no planner implementing this strategy has been published.

Conclusion

We return to the questions posed in our title.

Is reviving POCL planning possible? Likely not via the path most revival attempts have taken. The tradeoff we identified suggests why conceptually different delete-relaxation approaches failed in similar ways, effectively ruling out better engineering of such heuristics as a potential path forward. We believe that the question is not whether POCL planning is doomed, but rather whether the planning community abandoned it prematurely. Beyond delete-relaxation lies a largely unexplored design space: incremental heuristics that exploit refinement structure, modern heuristic families never transferred to POCL, and hybrid strategies that preserve structural advantages while controlling node growth. Our polynomial-time causal link compilation (Proposition 1) provides a foundation for such approaches, and advances in solver technology may alter what is practical.

Is reviving POCL planning worth it? We argue yes, especially for reasons beyond classical plan generation. HTN and temporal planners already rely on POCL structures (Bit-Monnot 2023; Bit-Monnot and Godet 2025) but lack the heuristic guidance available to state-based planners; stronger POCL heuristics would transfer directly. When concurrent execution is desired, POCL plans offer shorter makespans than parallel plans (Oates and Bercher 2026b), and their explicit causal structure supports plan explanation and analysis (Lindner and Olz 2022). Even closing the performance gap, without surpassing state-based methods, would benefit these downstream applications substantially.

A decade of neglect is not the same as a decade of evidence. POCL planning may yet have its revival, but it will require fresh perspectives on approaches that were promising but incomplete.

Acknowledgments

We sincerely thank all reviewers for helping to strengthen our arguments. 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

Bäckström, C. 1998. Computational aspects of reordering plans. *JAIR*, 9: 99–137.

Bäckström, C.; and Jonsson, P. 2011. All PSPACE-Complete Planning Problems Are Equal but Some Are More Equal than Others. In *SoCS 2011*, 10–17. AAAI Press.

Bercher, P. 2021. A Closer Look at Causal Links: Complexity Results for Delete-Relaxation in POCL Planning. In *ICAPS 2021*, 36–45. AAAI Press.

Bercher, P.; Geier, T.; and Biundo, S. 2013. Using State-Based Planning Heuristics for Partial-Order Causal-Link Planning. In *KI 2013*, 1–12. Springer.

Bercher, P.; Geier, T.; Richter, F.; and Biundo, S. 2013. On Delete Relaxation in Partial-Order Causal-Link Planning. In *ICTAI 2013*, 674–681. IEEE Computer Society.

Bercher, P.; Haslum, P.; and Muise, C. 2024. A Survey on Plan Optimization. In *IJCAI 2024*, 7941–7950. IJCAI.

Bercher, P.; Lin, S.; and Alford, R. 2022. Tight Bounds for Hybrid Planning. In *IJCAI-ECAI 2022*, 4597–4605. IJCAI.

Bercher, P.; and Olz, C. 2020. POP $\equiv$ POCL, right? Complexity Results for Partial Order (Causal Link) Makespan Minimization. In *AAAI 2020*, 9785–9793. AAAI Press.

Bit-Monnot, A. 2023. Experimenting with Lifted Plan-Space Planning as Scheduling: Aries in the 2023 IPC. In *IPC 2023*.

Bit-Monnot, A.; Ghallab, M.; Ingrand, F.; and Smith, D. E. 2020. FAPE: a constraint-based planner for generative and hierarchical temporal planning. *CoRR*, abs/2010.13121.

Bit-Monnot, A.; and Godet, R. 2025. Towards Canonical and Minimal Solutions in a Constraint-Based Plan-Space Planner. In *ECAI 2025*, 4678–4685. IOS Press.

Blum, A. L.; and Furst, M. L. 1995. Fast Planning Through Planning Graph Analysis. In *IJCAI 1995*, 1636–1642. Morgan Kaufmann.

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

Bonet, B.; Loerincs, G.; and Geffner, H. 1997. A robust and fast action selection mechanism for planning. In *AAAI 1997*, 714–719. AAAI Press.

Bylander, T. 1997. A Linear Programming Heuristic for Optimal Planning. In *AAAI 1997*, 694–699. AAAI Press.

Coles, A.; Coles, A.; Fox, M.; and Long, D. 2010. Forward-Chaining Partial-Order Planning. In *ICAPS 2010*, 42–49. AAAI Press.

Fikes, R. E.; and Nilsson, N. J. 1971. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artif. Intell.*, 2: 189–208.

Haslum, P.; Botea, A.; Helmert, M.; Bonet, B.; and Koenig, S. 2007. Domain-Independent Construction of Pattern Database Heuristics for Cost-Optimal Planning. In *AAAI 2007*, 1007–1012. AAAI Press.

Helmert, M.; and Domshlak, C. 2009. Landmarks, Critical Paths and Abstractions: What's the Difference Anyway? In *ICAPS 2009*, 162–169. AAAI Press.

Helmert, M.; Haslum, P.; and Hoffmann, J. 2007. Flexible Abstraction Heuristics for Optimal Sequential Planning. In *ICAPS 2007*, 176–183. AAAI Press.

Hoffmann, J.; and Nebel, B. 2001. The FF Planning System: Fast Plan Generation Through Heuristic Search. *JAIR*, 14: 253–302.

Kambhampati, S. 1997. Refinement Planning as a Unifying Framework for Plan Synthesis. *AI Mag.*, 18(2): 67–98.

Kambhampati, S.; Knoblock, C. A.; and Yang, Q. 1995. Planning as refinement search: a unified framework for evaluating design tradeoffs in partial-order planning. *Artif. Intell.*, 76(1): 167–238.

Khachiyan, L. G. 1979. A Polynomial Algorithm in Linear Programming. *Doklady Akademii Nauk SSSR*, 244(5): 1093–1096.

Koch, T.; Berthold, T.; Pedersen, J.; and Vanaret, C. 2022. Progress in Mathematical Programming Solvers from 2001 to 2020. *EURO Journal on Computational Optimization*, 10.

Lawler, E. L.; and Labetoulle, J. 1978. On Preemptive Scheduling of Unrelated Parallel Processors by Linear Programming. *J. ACM*, 25(4): 612–619.

Lindner, F.; and Olz, C. 2022. Step-by-Step Task Plan Explanations Beyond Causal Links. In *RO-MAN 2022*, 45–51. IEEE Press.

Lipovetzky, N.; and Geffner, H. 2012. Width and Serialization of Classical Planning Problems. In *ECAI 2012*, 540–545. IOS Press.

McDermott, D. V. 1996. A Heuristic Estimator for Means-Ends Analysis in Planning. In *AIPS 1996*, 142–149. AAAI Press.

Minton, S.; Bresina, J. L.; and Drummond, M. 1994. Total-Order and Partial-Order Planning: A Comparative Analysis. *JAIR*, 2: 227–262.

Nguyen, X.; and Kambhampati, S. 2001. Reviving Partial Order Planning. In *IJCAI 2001*, 459–464. Morgan Kaufmann.

Nguyen, X.; Kambhampati, S.; and Nigenda, R. S. 2002. Planning graph as the basis for deriving heuristics for plan synthesis by state space and CSP search. *Artif. Intell.*, 135(1): 73–123.

Oates, H.; and Bercher, P. 2026a. Are White Knights Worth the Trouble? Reconciling POCL and Partial-Order Plans for Plan Optimization. In *IJCAI-ECAI 2026*. IJCAI.

Oates, H.; and Bercher, P. 2026b. Makespan Investigations of Sequential, Parallel, PO, and POCL Plans. In *AAAI 2026*. AAAI Press.

Oates, H.; and Bercher, P. 2026c. Relaxing Is Hard: Complexity Results for Lifted Partial Order Causal Link Planning. In *ICAPS 2026*. AAAI Press.

Penberthy, J. S.; and Weld, D. S. 1992. UCPOP: A Sound, Complete, Partial Order Planner for ADL. In *KR 1992*, 103–114. Morgan Kaufmann.

Pommerening, F.; and Helmert, M. 2012. Optimal Planning for Delete-Free Tasks with Incremental LM-Cut. In *ICAPS 2012*, 363–367.

Pommerening, F.; Röger, G.; Helmert, M.; and Bonet, B. 2014. LP-Based Heuristics for Cost-Optimal Planning. In *ICAPS 2014*, 226–234. AAAI Press.

Sievers, S.; and Helmert, M. 2021. Merge-and-Shrink: A Compositional Theory of Transformations of Factored Transition Systems. *JAIR*, 71: 781–883.

Turner, H. 2002. Polynomial-Length Planning Spans the Polynomial Hierarchy. In *JELIA 2002*, 111–124. Springer.

Weld, D. S. 1994. An Introduction to Least Commitment Planning. *AI Mag.*, 15(4): 27–61.

Weld, D. S. 1999. Recent advances in AI planning. *AI Mag.*, 20(2): 93–93.

Weld, D. S. 2011. Systematic Nonlinear Planning: A Commentary. *AI Mag.*, 32(1): 101–103.

Younes, H. L.; and Simmons, R. G. 2002. On the Role of Ground Actions in Refinement Planning. In *AIPS 2002*, 54–62. AAAI Press.

Younes, H. L.; and Simmons, R. G. 2003. VHPOP: Versatile Heuristic Partial Order Planner. *JAIR*, 20: 405–430.