Why Not? Solver-Grounded Certificates for Explainable Mission Planning

Anonymous CVPR submission

Paper ID *****

Abstract

Operators of Earth observation satellites need justifications for scheduling decisions: why a request was selected, rejected, or what changes would make it schedulable. Existing approaches construct post-hoc reasoning layers independent of the optimizer, risking non-causal attributions, incomplete constraint conjunctions, and solver-path dependence. We take a faithfulness-first approach: every explanation is a certificate derived from the optimization model itself: minimal infeasible subsets for rejections, tight constraints and contrastive trade-offs for selections, and inverse solves for what-if queries. On a scheduling instance with structurally distinct constraint interactions, certificates achieve perfect soundness with respect to the solver's constraint model (15/15 cited-constraint checks), counterfactual validity (7/7), and stability (Jaccard = 1.0 across 28 seed-pairs), while a post-hoc baseline produces non-causal attributions in 29% of cases and misses constraint conjunctions in every multi-cause rejection. A scalability analysis up to 200 orders and 30 satellites confirms practical extraction times for operational batches.

1. Introduction

Earth observation (EO) satellite scheduling combines imaging task selection with downlink planning under coupled resource constraints. Recent surveys identify integrated imaging and downlink planning as a core class of satellite scheduling problems [5]. As these systems become more autonomous, operators need transparency into scheduling decisions. This transparency supports situational awareness and trust. Without it, operators cannot verify that the scheduler respects mission priorities, and they lack recourse when the schedule conflicts with operational judgment.

The explainability challenge centers on three query types: *why* a request was selected, *why not* when it is rejected, and *what if* to find changes that would make a rejected request schedulable. All three must account for the coupling between imaging and downlink, since an imaging decision is only viable if a later downlink pass can retrieve the data before storage overflows.

Existing work emphasizes presentation mechanisms [13–15]. These include argumentation structures, knowledge graphs, and natural language interfaces. While they improve operator interaction, they lack formal guarantees that explanations accurately reflect the underlying optimization model.

This gap manifests in three concrete failure modes: (1) *non-causal attribution*, where the explanation cites a plausible but non-decisive factor (e.g., storage pressure when the actual cause is a missing downlink); (2) *incomplete conjunctions*, where only a subset of jointly necessary constraint kinds is reported, leading to corrections that have no effect; and (3) *solver-path dependence*, where explanations change when the solver is re-run with a different random seed. We formalize these in Section 4.1 and demonstrate each empirically in Section 5.

This paper addresses the gap between explanation interfaces and solver behavior. We take a faithfulness-first approach. We define explanation faithfulness as the property that every claim in an explanation corresponds to a verifiable statement over the scheduler's constraints and decision variables. Each explanation is a certificate: a compact subset of model constraints and variable bindings that can be independently checked against the optimization formulation. This approach builds on optimization infeasibility analysis [2, 8], which identifies minimal sets of constraints responsible for infeasibility, and on constraint satisfaction explanation [7, 9], which provides efficient algorithms for computing such sets.

The contributions of this work are fourfold:

- An integrated EO scheduling formulation with imaging/downlink coupling via storage feasibility, designed for explanation with semantic tagging of constraint families that maps solver artifacts to operator-understandable categories.
- Certificate-based methods for why and why-not queries using minimal conflict sets and contrastive dominance analysis, with formal soundness guarantees.
- Counterfactual explanations for what-if queries via minimal corrections, validated through re-solving to confirm

the predicted schedule change.

- A faithfulness evaluation protocol with metrics for soundness, minimality, and stability, enabling regression testing as the scheduling model evolves.

2. Related Work

Explainability for satellite scheduling. Powell and Riccardi motivate explainability for onboard satellite scheduling [13]. They propose abstract argumentation to explain scheduling outcomes [14]. Later work uses knowledge graphs and question answering for interactive explanations [15]. They also explore natural language generation for operator-facing explanations [17]. Maillard *et al*. apply explainable scheduling to coverage analysis in constellation design [12]. These works show that transparency matters for operational acceptance.

Explainability in constraint satisfaction and optimization. Constraint reasoning provides a foundation for explainable decision support. Surveys consolidate methods for generating explanations from minimal conflicting subsets and correction subsets [9]. Step-wise frameworks produce interpretable and verifiable explanation sequences [1]. Efficient algorithms compute optimal unsatisfiable subsets for constraint satisfaction problems [6, 7].

Counterfactual explanations have been formalized through inverse optimization and inverse constraint programming [10, 11]. These produce minimal changes to make a desired decision feasible. They directly address why-not and what-if queries with verifiable answers. Outside space applications, argumentation-supported tools provide structured interaction with scheduling outcomes [4].

Gap. No existing space-specific approach grounds explanations in the solver's constraint model or provides formal correctness guarantees.

3. Problem Formulation

This section translates the physical and operational rules governing EO scheduling (visibility windows, onboard storage limits, ground-station contacts, and slew constraints) into a mixed-integer optimization model. The formulation is designed so that every constraint carries a semantic tag linking it back to an operator-understandable category, which is the basis for the certificate-based explanations in Section 4.

3.1. Integrated EO Scheduling

Consider an EO satellite system operating over a planning horizon $[0, T]$. The system comprises one or more spacecraft and a set of ground stations. The scheduler receives a set of imaging requests (orders) and must decide which to acquire, when, and how to downlink the resulting data. The objective is to maximize mission value subject to physical, operational, and policy constraints.

Decision variables. Let $\mathcal{O}$ denote the set of imaging orders and $\mathcal{P}$ the set of imaging passes (time windows during which a spacecraft can observe a target). Let $\mathcal{Q}$ denote the set of downlink passes (time windows during which a spacecraft is in contact with a ground station). The model uses four families of binary decision variables:

$$x_{o,p} \in \{0,1\} \quad \text{order } o \text{ assigned to imaging pass } p, \tag{1}$$
$$y_p \in \{0,1\} \quad \text{imaging pass } p \text{ is used}, \tag{2}$$
$$d_q \in \{0,1\} \quad \text{downlink pass } q \text{ is used}, \tag{3}$$
$$a_o \in \{0,1\} \quad \text{order } o \text{ is scheduled}. \tag{4}$$

An order is considered scheduled only if it is imaged and downlink resources are allocated such that delivery is feasible under onboard storage constraints. This coupling between imaging and downlink is central to the formulation and distinguishes it from imaging-only models common in the literature.

Objective function. The scheduler maximizes a weighted combination of mission value terms. Each order o has a base value V_o and a priority level P_o. The priority is mapped through a weight function $W(P_o) = 1 + \alpha(P_o - 1)$, where α is a tunable parameter that controls how steeply higher-priority orders are preferred. The objective includes a reward for scheduled orders, a penalty for downlink resource usage, a penalty for cloud risk, and a penalty for data latency:

$$\begin{aligned} \max \quad & \sum_{o,p} V_o\, W(P_o)\, x_{o,p} + \beta \sum_{o} a_o \\ & - \lambda \sum_{q} d_q - \mu \sum_{o,p} V_o\, W(P_o)\, c_p\, x_{o,p} \\ & - \eta \sum_{o,p} V_o\, W(P_o)\, \ell_p\, x_{o,p} \end{aligned} \tag{5}$$

where c_p is the fractional cloud-cover forecast for pass p, ℓ_p is the normalized latency from imaging to the earliest available downlink (computed from pass geometry, not from the decision variables), and $\beta, \lambda, \mu, \eta$ are scalar weights. The β term is set small relative to order values and acts as a tiebreaker, preferring schedules that include more orders when value alone is indifferent. The λ term discourages unnecessary downlink passes, since each ground-station contact has an operational cost. The cloud and latency penalties steer the scheduler toward clearer skies and faster data delivery when multiple imaging opportunities exist.

Assignment constraints. Each order is assigned to at most one imaging pass, and the pass-usage and order-scheduled indicators are linked to the assignment variables:

$$\sum_p x_{o,p} \le 1 \qquad \forall\, o \in \mathcal{O}, \tag{6}$$

$$x_{o,p} \le y_p \qquad \forall\, o, p, \tag{7}$$

$$a_o = \sum_p x_{o,p} \qquad \forall\, o \in \mathcal{O}, \tag{8}$$

$$y_p \le \sum_o x_{o,p} \qquad \forall\, p \in \mathcal{P}. \tag{9}$$

Constraint (6) ensures single assignment; together with binary domains, it implies $a_o \in \{0, 1\}$ via (8). Constraints (7) and (9) link pass usage to assignments so that a pass is marked as used if and only if at least one order is assigned to it.

Downlink requirement. An imaging pass is usable only if at least one subsequent downlink pass on the same spacecraft is selected. Let $\mathcal{D}(p)$ denote the set of downlink passes for the same spacecraft that begin after imaging pass p ends:

$$y_p \le \sum_{q \in \mathcal{D}(p)} d_q \quad \forall\, p \in \mathcal{P}. \tag{10}$$

If $\mathcal{D}(p) = \varnothing$, no subsequent downlink exists, and y_p is fixed to zero. This constraint encodes the requirement that acquired data must be retrievable. It is the primary source of imaging-downlink coupling: selecting an imaging pass forces the selection of a compatible downlink pass, which consumes ground-station time and affects storage dynamics.

Temporal exclusion. Passes on the same spacecraft that overlap in time or require an attitude slew exceeding the available transition time cannot both be selected. Let $\mathcal{E}$ denote the set of conflicting pass pairs (exclusions), constructed by checking temporal overlap and minimum slew duration. For each pair $(i, j) \in \mathcal{E}$, where u_i and u_j are the corresponding pass-selection variables (y for imaging, d for downlink):

$$u_i + u_j \le 1 \quad \forall\, (i, j) \in \mathcal{E}. \tag{11}$$

Storage reservoir. Onboard storage evolves as images are acquired and data is transmitted. For each spacecraft, passes are ordered chronologically. Let s_0 denote the initial storage level and C the storage capacity. At each pass k in the chronological sequence, the cumulative storage must remain within bounds:

$$0 \le s_0 + \sum_{j \le k} \Delta_j \le C \quad \forall\, k, \tag{12}$$

where $\Delta_j = \sum_o \text{data}_o\, x_{o,j}$ for an imaging pass (storage increases by the data volume of assigned orders) and $\Delta_j = -\text{tx}_j\, d_j$ for a downlink pass (storage decreases by the transmitted volume). Here tx_j is the data volume that downlink pass j can transmit, determined by the ground-station link rate and the pass duration. This reservoir constraint couples all imaging and downlink decisions on the same spacecraft into a single resource trajectory.

Feasibility filters. Before the optimization model is constructed, several feasibility checks prune the variable space. Orders are excluded from passes that occur after their deadline. Passes with cloud-cover forecasts exceeding a hard threshold are removed. Imaging passes are excluded when the spacecraft is unavailable (e.g., during maintenance or orbit maneuvers). Downlink passes are excluded when the ground station is unavailable. These filters reduce model size without affecting optimality, since the removed assignments are infeasible by definition.

3.2. Semantic Constraint Tags

Each constraint in the model is annotated with a semantic tag drawn from a fixed taxonomy. The tags identify the physical or operational category of the constraint: VISIBILITY, DEADLINE, CLOUD, STORAGE, DOWNLINK, TEMPORAL, ENERGY, and POLICY. These tags serve two purposes. First, they allow explanation algorithms to project solver-internal constraint identifiers onto categories that operators understand. An unsatisfiable core expressed as a set of constraint indices is not useful to an operator. The same core expressed as "storage capacity at time t_3 plus downlink requirement for pass p_7" is actionable. Second, the tags enable aggregation. When a minimal conflict contains multiple storage constraints at consecutive time steps, the tag allows grouping them into a single "storage trajectory conflict" rather than listing each bound separately.

3.3. Faithfulness Requirements

Explanation faithfulness means that every claim in an explanation is a verifiable statement over the optimization model and its solution. We formalize this through three properties, each designed to prevent the failure modes identified in Section 4.1.

Soundness requires that each constraint cited in an explanation is indeed a constraint of the model and that the claimed status is verifiable. For infeasibility certificates, soundness has two levels: the cited set must be *jointly sufficient* to prove infeasibility (set-level check), and each individual constraint must be *necessary*, meaning that removing it yields a feasible witness in which that constraint is violated (per-constraint check).

Minimality requires that no proper subset of the cited constraints suffices to establish the explanation. For a why-

not explanation, this means the conflict set is irreducible: removing any single constraint makes the augmented problem (with a forced in) feasible. Minimality prevents explanations from including extraneous constraints that obscure the decisive factors.

Completeness requires that the explanation accounts for all necessary conditions. For a why-not explanation based on infeasibility, completeness means the cited constraints are not only individually necessary but jointly sufficient to establish infeasibility. This aligns with the notion of irreducible infeasible subsystems in mathematical programming [2, 3, 8].

4. Proposed Approach

Figure 1 illustrates the explanation pipeline. An operator poses a query about a specific order: why it was selected, why it was rejected, or what would make it schedulable. The system augments the scheduling model to reflect the query (e.g., forcing a rejected order into the schedule), runs the solver or an infeasibility analysis on the augmented model, and extracts a certificate: a compact, verifiable subset of constraints and variable bindings that answers the query. Semantic tags (Section 3.2) then project the certificate onto operator-understandable categories. Presentation layers (argumentation frameworks, knowledge graphs, natural language interfaces) may rephrase or reorder [1] the certificate but must not add claims beyond it. We first identify the failure modes that motivate this approach, then detail each certificate type.

4.1. Challenges with Post-hoc Explanation Layers

Explanation layers built separately from the optimizer can diverge from the true causes encoded in the model. The three failure modes identified in Section 1 (non-causal attribution, incomplete conjunctions, and solver-path dependence) motivate the solver-grounded certificates detailed below. Section 5 demonstrates each on a realistic scheduling instance.

4.2. Certificate-Based Explanations for Why-Not Queries

When an operator queries why order a is not in the schedule S^*, we construct an augmented problem M_a by adding the constraint $a_a = 1$ (forcing order a to be scheduled) to the original model M. Two cases arise.

Case 1: M_a is infeasible. No feasible schedule exists that includes a under the current constraints. The explanation is a minimal infeasible subset (MIS) of the constraints in M_a. A MIS is a subset $I \subseteq M_a$ such that I is infeasible and every proper subset of I is feasible. This is also known as an irreducible infeasible subsystem (IIS) in the mathematical programming literature [2, 8].

To compute the MIS, we use a deletion-based algorithm that isolates the blocking constraints by testing which ones can be removed without restoring feasibility. Starting from M_a (which is infeasible by assumption), we iterate over each constraint c in M_a. We temporarily remove c and solve the relaxed problem. If the relaxed problem is still infeasible, c is redundant for this particular infeasibility and is permanently removed. If the relaxed problem becomes feasible, c is necessary and is retained. After processing all constraints, the remaining set is a MIS, the smallest set of rules that together make scheduling order a impossible.

The cost is at most $|M_a|$ feasibility checks (no objective needed). For models with large constraint counts but small cores, grow-shrink variants [3] can be faster than deletion from the full set.

Once the MIS is computed, the semantic tags from Section 3.2 project it onto operator-understandable terms. For example, a MIS containing a downlink requirement constraint (10) for pass p_7, a temporal exclusion constraint (11) for passes (p_4, p_7), and the forcing constraint $a_a = 1$ is reported as: "Order a requires imaging pass p_7, which needs a downlink afterward, but p_7 conflicts temporally with pass p_4 (serving order b). No alternative imaging pass for a has a viable downlink."

Case 2: M_a is feasible but a is not selected. A feasible schedule including a exists, but the optimizer did not choose it because including a reduces the objective. Let S_a^* denote the optimal schedule with a forced in. The explanation is a contrastive certificate: the set of orders in S^* that are displaced in S_a^*, together with the objective difference $\text{obj}(S^*) - \text{obj}(S_a^*)$. This directly answers the operator's question: including a would require dropping orders $\{b_1, b_2\}$ and would reduce mission value by δ.

To ensure this explanation is minimal, we check whether forcing a while retaining each displaced order individually is feasible. If displacing only b_1 suffices (i.e., forcing both a and b_2 is feasible), then b_2 is not a necessary displacement and is removed from the explanation.

4.3. Certificate-Based Explanations for Why Queries

For a scheduled order $a \in S^*$, the operator wants to know why a was included. The explanation has two parts.

Tight constraints. We identify constraints from the model that are binding (satisfied with equality) at S^* and involve the decision variables of order a. For example, if the storage reservoir constraint (12) is tight at the time step immediately after a's imaging pass, this indicates that a was scheduled under storage pressure and that any additional imaging at that point would overflow the buffer. Similarly, if the temporal exclusion constraint (11) for the pair (p_a, p_c)

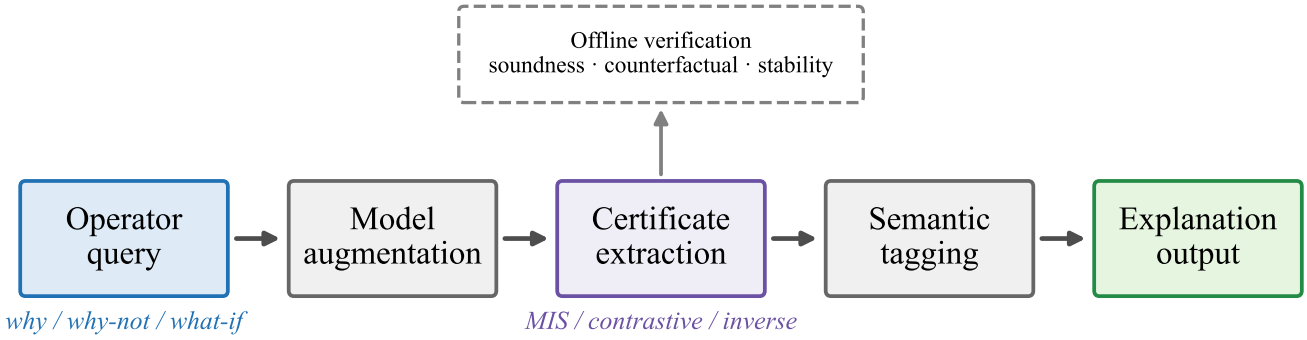

Figure 1. Explanation pipeline. An operator query triggers model augmentation and certificate extraction (MIS for why-not, contrastive analysis for why, inverse solve for what-if). Semantic tags map solver constraints to operator categories. Offline verification (dashed) is not required at query time.

is binding, it shows that a's imaging pass directly excluded pass p_c.

Contrastive dominance. We identify alternative orders that could have been scheduled in place of a but were not. For each such alternative b, we solve the problem with a excluded and b forced in. If the resulting objective is lower, a dominates b in this context, and the explanation reports the value difference. If no feasible schedule exists with b in place of a, the explanation reports that b is not a viable alternative. This gives the operator a concrete sense of the trade-offs the scheduler made.

4.4. Counterfactual Explanations via Minimal Corrections

A why-not certificate tells the operator *what* is blocking a rejected order, but not *what to change* to unblock it. For what-if queries, the operator wants to know the smallest modification to the mission plan that would make a rejected order a schedulable. This completes the explanation loop: the MIS identifies the decisive constraints, and the minimal correction identifies the cheapest way to relax them. We formalize this as an inverse problem following [10, 11].

Change space. Not all model parameters are under the operator's control. We define a set of operator-actionable parameters Θ that reflects the levers available in practice: cloud-cover thresholds (relaxing from 20% to a higher value, accepting more imaging risk), ground-station availability windows (adding a contact pass by coordinating with station operators), storage reserve margins (reducing the safety buffer to free capacity), and priority weights (increasing P_a to make a more competitive against other orders). Each parameter $\theta_i \in \Theta$ has a current value θ_i^0 and a cost $w_i(|\theta_i - \theta_i^0|)$ that measures the operational cost of deviation. These costs encode domain knowledge: cloud-threshold relaxations are inexpensive (the operator simply accepts more risk), while adding ground-station passes is expensive (it requires coordination with external operators and may displace other missions' contacts).

Inverse problem. Given the change space, we seek the cheapest set of parameter modifications that makes a schedulable:

$$\min_{\Delta \in \Theta} \sum_i w_i(|\Delta_i|) \quad \text{s.t.} \quad M(\theta^0 + \Delta) \text{ is feasible with } a_a = 1, \tag{13}$$

where $M(\theta)$ denotes the scheduling model parameterized by θ. The solution Δ^* is a minimal correction: the cheapest set of parameter changes that makes a schedulable. When the change space is discrete and small (e.g., a few candidate additional ground-station passes), this reduces to enumerating combinations. When it is continuous (e.g., cloud thresholds), the problem can be formulated as a mixed-integer program with the correction magnitudes as additional variables.

Validation. The inverse formulation operates on a parameterized version of the model, which may introduce approximation gaps relative to the full scheduling problem. We therefore validate each minimal correction by re-solving the complete scheduling model $M(\theta^0 + \Delta^*)$ with $a_a = 1$ and verifying that the resulting schedule indeed includes a with an objective value above the operator's threshold. This resolve step guards against cases where the correction is necessary but not sufficient in the full model, and confirms that the operator can trust the recommendation.

Reporting. The output to the operator states the correction in operational terms: "Order a becomes schedulable if the cloud-cover threshold for pass p_3 is relaxed from 20% to 28%, or if an additional downlink pass is added at ground station G_2 between 14:00 and 14:12 UTC." When multiple minimal corrections exist, we report the one with lowest operational cost and optionally the full Pareto set of cost-equivalence classes. This gives the operator a concrete menu of interventions ranked by feasibility, rather than a single take-it-or-leave-it recommendation.

5. Experimental Evaluation

We evaluate certificate-based explanations along two axes. First, we verify *correctness*: are the certificates sound, minimal, counterfactually valid, and stable across solver runs? We then compare against a representative post-hoc baseline to quantify the failure modes from Section 4. Second, we characterize *computational cost* across a range of problem sizes to assess practical scalability.

5.1. Experimental Setup

Scheduling model and orbital mechanics. We instantiate the integrated imaging-downlink model from Section 3 using a constellation of three commercial synthetic aperture radar (SAR) satellites with real two-line element sets (TLEs): two spacecraft (S1, S2) at 53° inclination and one (S3) in a 97.7° sun-synchronous orbit. All have 8.0 GB onboard storage, 15.0 Mbps downlink rate, and 150 s minimum slew time. Ten imaging orders are placed on verified ground tracks over a 12-hour simulation window. Imaging and downlink passes are generated via Skyfield/SGP4 propagation.

Test instance. We construct a single instance that exercises structurally distinct constraint interactions by restricting downlink access to a single ground station (Svalbard). Because S1 and S2 fly at 53° inclination, they cannot reach the polar station and receive *zero* downlink passes. Only the sun-synchronous S3 retains downlink access (4 passes). Initial storage is 80% (1.6 GB free); each order requires 1.8 GB. The model contains 38 passes (34 imaging, 4 downlink) and 110 semantically tagged constraints.

This design introduces a structural asymmetry: orders with imaging candidates on both satellite types encounter *different* blocking constraints on each path. Candidates on S1/S2 are blocked by missing downlink (NO_DOWNLINK), while candidates on S3 are blocked by storage overflow (STORAGE_UPPER_BOUND).

Of 10 orders, 1 is scheduled, 2 are rejected by optimality trade-offs (schedulable but displaced by higher-priority orders), and 7 are constraint-infeasible, yielding 7 infeasibility certificates.

Post-hoc baseline. The baseline implements the optimizer-agnostic explanation pattern shared by all existing satellite scheduling explanation work (argumentation frameworks [14], knowledge graphs [15], and LLM-generated text [16]), which operate as post-hoc layers without access to solver internals. For each unscheduled order, the baseline identifies candidate imaging passes, inspects the schedule state (temporal conflicts, downlink availability, storage trajectory), and reports blocking reasons for the single best candidate. It uses the same semantic categories as the certificate approach but derives them from schedule-state inspection. Crucially, it evaluates each candidate independently and selects the one with the fewest blocking reasons, so it cannot reason about the joint constraint structure across all candidates. A multi-candidate variant could inspect all candidates and reduce non-causal attributions, but it would still lack the ability to prove that two constraint kinds are *jointly* necessary, since that requires reasoning over the solver's constraint model across all candidates simultaneously.

5.2. Correctness and Baseline Comparison

We now test whether the certificates correctly identify the true bottlenecks in the Svalbard scenario, specifically missing downlink for the inclined-orbit satellites and storage overflow for the sun-synchronous one, and whether the post-hoc baseline misses any.

Intrinsic correctness. Soundness and stability are algorithmic guarantees by construction of the MIS algorithm; we verify them here concretely. The 7 infeasibility certificates cite 15 constraints in total. We check each against the three soundness conditions from Section 3 (tag validity, collective sufficiency, and individual necessity); all 15 pass all three checks (Table 1). The certificates are also minimal, averaging 2.1 constraints (median 2.0, maximum 3), well within operator working memory. To confirm counterfactual validity, we derive a targeted correction from each certificate's constraint kinds (e.g., adding storage capacity when STORAGE_UPPER_BOUND is cited, inserting a synthetic downlink pass when NO_DOWNLINK is cited), apply it to a cloned instance, and re-solve; all 7 previously rejected orders become schedulable. Finally, re-running the instance with 8 different solver seeds yields identical explanation sets (Jaccard = 1.0 across all $\binom{8}{2} = 28$ seed pairs), confirming that certificates are stable with respect to solver non-determinism.

Baseline failure modes. We now compare the two approaches on the 7 infeasibility explanations, i.e. the cases where an order is rejected because no feasible schedule can include it. Using the certificate output as ground truth, we examine where the post-hoc baseline diverges. Of the three

Table 1. Evaluation results. All metrics are over the 7 constraint-infeasible orders unless noted. Top: intrinsic correctness. Bottom: post-hoc baseline failure modes (certificate output as ground truth).

Intrinsic correctness		
Infeasibility certificates	7	
Soundness checks passed	15 / 15	
Counterfactual tests succeeded	7 / 7	
Stability (Jaccard, 28 seed-pairs)	1.000	
Core size (avg / median / max)	2.1 / 2.0 / 3	
Post-hoc baseline failure modes		
Orders with non-causal attribution	2 / 7	(29%)
Non-causal / total attributions	2 / 9	(22%)
Conjunction orders ($\lvert G\rvert > 1$)	3	
Baseline incomplete on conjunctions	3 / 3	(100%)

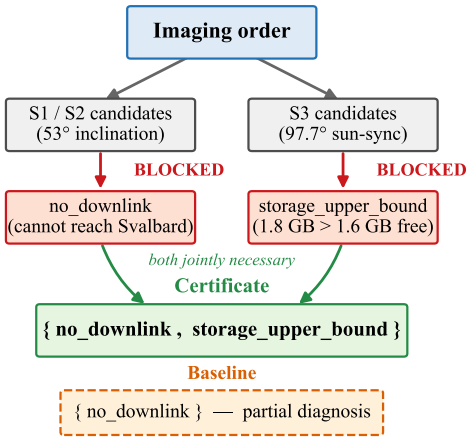

Figure 2. Conjunction failure mode. An order's candidates span two satellite types, each blocked by a different constraint kind. The certificate identifies both as jointly necessary; the baseline reports only the single best candidate's cause.

failure modes from Section 1, solver-path dependence is addressed by the stability result above (the baseline was not designed to be re-run across seeds). The remaining two, non-causal attribution and incomplete conjunctions, both manifest on this instance.

The first is *non-causal attribution*. Orders ORD-03 and ORD-04 are each blocked by storage overflow alone, but the baseline also cites NO_DOWNLINK because the candidate satellites lack downlink access. The missing downlink is a real constraint in the model, but it is not *necessary* for the proof of infeasibility: storage overflow would still block these orders even if downlink were available. An operator acting on the non-causal diagnosis (for example, by requesting an additional ground-station contact) would undertake an expensive and unnecessary intervention. In total, 2 of 7 orders (29%) receive at least one non-causal attribution (Table 1).

The second failure mode is *incomplete conjunctions*, and it is the more consequential one. Three orders (ORD-01, ORD-06, ORD-09) have imaging candidates on two satellite types: the 53°-inclination spacecraft (blocked by missing downlink to Svalbard) and the sun-synchronous spacecraft (blocked by storage overflow, since 1.8 GB exceeds 1.6 GB free). The certificate correctly identifies both NO_DOWNLINK and STORAGE_UPPER_BOUND as jointly necessary; removing either makes the order schedulable via the other satellite type. The baseline, however, evaluates candidates independently and selects the one with the fewest blocking reasons, a 53°-inclination pass blocked solely by missing downlink. It never discovers that storage overflow is jointly necessary. This occurs on all three conjunction orders (3/3), meaning an operator would receive an incomplete diagnosis every time the true cause spans multiple constraint kinds (Figure 2).

Table 2. Wall-clock timing breakdown for the test instance (3 satellites, 1 ground station, 10 orders, Apple M4 CPU).

Component	Time
Operational (query-time)	
Scheduling solve (CP-SAT)	6 ms
Certificate extraction (MIS)	440 ms
Offline validation	
Soundness verification	64 ms
Counterfactual testing	1 197 ms

5.3. Computational Cost

Having established correctness, we now ask whether certificate extraction is fast enough for operational use. Table 2 reports wall-clock times for the test instance, solved with Google's CP-SAT constraint-programming solver on an Apple M4 CPU. The operational cost (what a mission planner experiences at query time) comprises the scheduling solve and certificate extraction; soundness verification and counterfactual testing are offline validation steps run during development. For 7 certificates, the solve completes in 6 ms and certificate extraction takes 440 ms, giving sub-second total latency. The dominant cost is MIS extraction, which makes $O(k \cdot c)$ solver calls per infeasible order (k = core size, c = candidate constraints). Not all unscheduled orders require MIS extraction: pre-filtered orders (no visibility, cloud cover) and optimality trade-offs are resolved without the deletion algorithm, and each order's extraction is independent and trivially parallelizable.

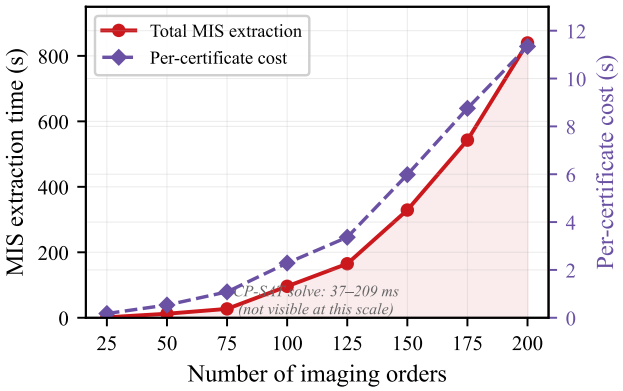

Figure 3. Operational latency vs. order count (10 satellites, 5 ground stations). MIS extraction dominates total latency; the CP-SAT solve remains negligible (<210 ms) across all tested sizes.

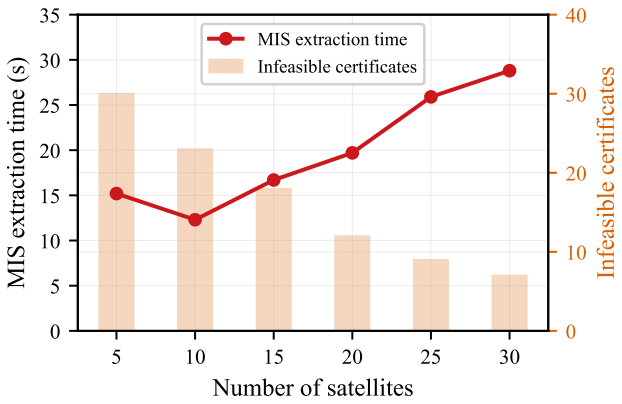

Figure 4. Constellation scaling (50 orders, 5 ground stations). Per-certificate cost increases with constellation size, but total latency remains bounded because more orders become schedulable, reducing the number of certificates needed (bars).

Scalability analysis. The test instance is small by design; to characterize how costs grow, we generate synthetic instances along two axes. First, we scale the number of orders from 25 to 200 in uniform increments (10 satellites, 5 ground stations). As shown in Figure 3, per-certificate cost rises from 176 ms at 25 orders to 11.3 s at 200 orders, because larger order sets produce more constraints for the deletion algorithm to iterate over. Total latency reaches 14 min at 200 orders. Second, we vary constellation size from 5 to 30 satellites with a fixed 50-order workload (Figure 4). Larger constellations schedule more orders (7 of 50 at 5 satellites vs. 31 at 30), which reduces the number of infeasible certificates from 30 to 7. Per-certificate cost still grows with model size (508 ms to 4.1 s), but total operational latency remains under 30 s because fewer orders require certificates. For typical operational batches (25–50 orders), certificate extraction completes in under 15 s, and larger batches can be parallelized since each order's extraction is independent.

5.4. Summary and Limitations

The results show that solver-grounded certificates are both correct and compact: perfect scores on all four intrinsic metrics, with cores small enough (2–3 constraints) for an operator to act on directly. The post-hoc baseline, by contrast, produces non-causal attributions in 29% of cases and fails to identify all blocking constraint kinds in every conjunction case. These are not edge cases; they arise from a structural limitation of optimizer-agnostic reasoning that no amount of candidate inspection can fully resolve. Certificate extraction adds modest overhead (sub-second for the test instance, under 15 s for typical batches) and is trivially parallelizable.

Several limitations scope these findings. Correctness is verified on a single structured instance; the scalability experiments are synthetic and measure runtime only. The baseline represents the architectural limitation of optimizer-agnostic approaches; stronger implementations that inspect multiple candidates could partially mitigate non-causal attribution, though not incomplete conjunctions. The counterfactual tests validate that corrections derived from certificate constraint kinds restore schedulability, but do not exercise the full inverse optimization pipeline (Equation 13) with operational cost weighting. User studies with mission operators are needed to validate explanation utility in practice.

6. Conclusion

This paper presents a faithfulness-first approach to explainable AI for integrated Earth observation scheduling. Explanations are grounded in verifiable solver certificates: minimal conflicts and minimal corrections. This ensures alignment between explanations and the actual optimization model. The evaluation confirms full correctness on all four intrinsic metrics with respect to the solver's constraint model and exposes concrete failure modes, non-causal attribution (29%) and incomplete conjunctions (100%), in the post-hoc baseline that is standard in the literature. A scalability analysis shows that certificate extraction remains practical for typical operational batches and is trivially parallelizable. Future work will implement these methods on operational EO instances and evaluate explanation utility through user studies with mission operators.

References

[1] Bart Bogaerts, Emilio Gamba, and Tias Guns. A framework for step-wise explaining how to solve constraint satisfaction problems. *Artificial Intelligence*, 2021. 2, 4

[2] John W. Chinneck. Finding a useful subset of constraints for analysis in an infeasible linear program. *INFORMS Journal on Computing*, 1997. 1, 4

[3] John W. Chinneck. Analyzing infeasible optimization models. Tutorial, CPAIOR, 2007. 4

[4] Kristijonas Cyras, Myles Lee, and Dimitrios Letsios. Schedule explainer: An argumentation-supported tool for interactive explanations in makespan scheduling. In *Explainable and Transparent AI and Multi-Agent Systems*, 2021. 2

[5] Benedetta Ferrari, Jean-François Cordeau, Maxence Delorme, Manuel Iori, and Roberto Orosei. Satellite scheduling problems: A survey of applications in earth and outer space observation. *Computers & Operations Research*, 173: 106875, 2025. 1

[6] Emilio Gamba, Bart Bogaerts, and Tias Guns. Efficiently explaining csps with unsatisfiable subset optimization. In *IJCAI*, pages 1371–1377, 2021. 2

[7] Emilio Gamba, Bart Bogaerts, and Tias Guns. Efficiently explaining csps with unsatisfiable subset optimization. *Journal of Artificial Intelligence Research*, 2023. 1, 2

[8] John Gleeson and Jennifer Ryan. Identifying minimally infeasible subsystems of inequalities. *INFORMS Journal on Computing*, 1990. 1, 4

[9] Sharmi Dev Gupta, Begum Genc, and Barry O'Sullivan. Explanation in constraint satisfaction: A survey. In *IJCAI*, pages 4400–4407, 2021. 1, 2

[10] Anton Korikov and J. Christopher Beck. Counterfactual explanations via inverse constraint programming. In *International Conference on Principles and Practice of Constraint Programming (CP)*, 2021. 2, 5

[11] Anton Korikov, Alexander Shleyfman, and J. Christopher Beck. Counterfactual explanations for optimization-based decisions in the context of the gdpr. In *IJCAI*, pages 4097–4102, 2021. 2, 5

[12] Adrien Maillard, Christopher Wells, Shadi Eveisgharan, Paul Rosen, and Steve Chien. Where is my coverage? using explainable automated scheduling to inform mission design of an earth-observing constellation. In *International Workshop on Planning and Scheduling for Space (IWPSS)*, 2023. 2

[13] Cheyenne Powell and Annalisa Riccardi. Towards explainability of on-board satellite scheduling for end user interactions. In *72nd International Astronautical Congress (IAC)*, Dubai, United Arab Emirates, 2021. 1, 2

[14] Cheyenne Powell and Annalisa Riccardi. Abstract argumentation for explainable satellite scheduling. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10, 2022. 2, 6

[15] Cheyenne Powell and Annalisa Riccardi. Question answering over knowledge graphs for explainable satellite scheduling. *Journal of Aerospace Information Systems*, 2025. 1, 2, 6

[16] Cheyenne Powell and Annalisa Riccardi. Generating textual explanations for scheduling systems leveraging large language models. *Journal of Intelligent Information Systems*, 2025. 6

[17] Cheyenne Powell, Audrey Berquand, and Annalisa Riccardi. Natural language processing for explainable satellite scheduling. In *SPACEOPS*, 2023. 2