APPENDIX

A. Completeness of the INFORMED algorithm

The algorithm described above inherits many of the same theoretical guarantees as those described in [1]. In particular, the proof of semi-completeness for optimistic algorithms (i.e. theorem 3) can be extended to the INFORMED algorithm under some assumptions about the relevance scores.

At a high level, the proof for theorem 3 argues that if $\tilde{l}$ is the minimum level at which a solution exists, a finite number of iterations will elapse before the algorithm considers all stream instances with level $l \leq \tilde{l}$, at which point the solution must be found.

The main point of departure between these optimistic algorithms and INFORMED is the use of predicted relevance scores, rather than level (equation 1) for ordering the instantiation of streams. Therefore, to extend the argument to INFORMED, we simply need to show that there is a finite number of stream instances with a score greater than y for any $y > 0$.

Theorem 1.1: The INFORMED algorithm is semi-complete under the following assumptions: (1) that the score that $\mathcal{M}_\theta$ assigns to a stream is strictly less than the scores assigned to its parents, and (2) that the relevance scores assigned to successive evaluations of the same stream instance are monotonically decreasing.

Proof: Let π be a feasible solution to a given PDDLStream problem. Then PREIMAGE(π) is the set of stream-certified facts which must be added the initial conditions $\mathcal{I}$ so that SEARCH may find the correct optimistic solution. Let y^* be the minimum score assigned to any of the elements of PREIMAGE(π).

We can show that under conditions (1) and (2), a stream with score $\geq y^*$ has a maximum possible level, as defined in equation 1. To see this, let $\epsilon_{child} > 0$ be the minimum difference of scores between a stream and its child, and let $\epsilon_{eval} > 0$ be the minimum difference between successive evaluations of the same stream. We can bound the maximum level of a stream with score y^* as $\max_l(y^*) \leq 1 + \lceil \frac{1-y^*}{\epsilon_{eval}} \rceil + \max_l(y^* + \epsilon_{child}) \leq \frac{1-y^*}{e_{child}}(1 + \frac{1-y^*}{e_{eval}})$.

INFORMED will instantiate the stream with the highest score at every iteration. By the argument above, every stream with score $\geq y^*$ must have level $\leq \max_l(y^*)$. By theorem 3, we know that the number of stream instances with level $\leq max_l(y^*)$ is finite. Therefore, we conclude that a finite number of iterations will elapse before the algorithm considers all stream instances with score $\geq y^*$, at which point a solution must be found. ■

In practice we enforce condition (2) by decaying the relevance score by a factor $\gamma < 1$ after every evaluation. Condition (1) is enforced by defining the score of a stream s as:

$$score(s) = M_\theta(s) \cdot \min_{\hat{s} \in s.parents} score(\hat{s})$$

During training, we approximate the min with a weighted average of the parent scores, where the weights are computed as a softmax over the negative scores.

B. Model Architecture and Hyperparameters

In section IV-C, we have outlined the representation of planning problems as a relational graph, augmented with features that encode the 3D position of each object. This Problem Graph is the input to a GNN, composed of 3 graph network (GN) blocks [27]. Each GN block is itself composed of a node model ϕ_v, and an edge model ϕ_e which share the same architecture in table IV.

hidden size	64
output size	64
number of layers	2
hidden activation	LeakyRelu
output activation	Linear

TABLE IV: Hyperparameters of node and edge encoders $\phi^{[1/2/3]}_{[v/e]}$ which comprise each of 3 GN blocks.

While this GNN produces embeddings of both its nodes and its edges as output, we only use the node embeddings, which serve as the inputs to the stream MLPs. Each stream MLP has a different number of inputs I_s and outputs O_s (depending on the domain definitions), however each of the input and output objects will be represented by a fixed size embedding. We detail the architecture of each MLP in V.

	$M_s^{encoder}$	M_s^{scorer}	$M_s^{decoder}$
input size	$64 \cdot I_s$	64	64
number of layers	1	2	2
hidden size	64	64	64
hidden activation	-	LeakyRelu	LeakyRelu
output size	64	1	$64 \cdot O_s$

TABLE V: Architecture and hyperparameters for each stream MLP. Note that the size of the input to the encoder, and output of the decoder depend on the stream's domain definitions.

C. Problem Distribution Details

TABLE VI: To supplement the descriptions of environments given in section V, we provide details of object numbers and placement distributions for the train/test sets.

	Number of Objects Train / Test	**Placement Distribution**
Clutter	Blocks: 2-4 / 2-6 Blockers: 4-8 / 4-12	Poisson Disc
Non Monotonic	Blocks: 1-3 / 2-6 Blockers: 1-3 / 2-6	Uniform Adjacent
Sorting	Blocks: 2-7 / 2-10 Blockers: 2-7 / 2-10	Poisson Disc
Stacking	Blocks: 2-4 / 2-7 Blockers: 0 / 0	Uniform
Distractors	Blocks: - / 2-3 Blockers: - / 10-50	Uniform(red, blue) Uniform(green, purple)
Kitchen	Cabbage: 1-4 / 1-3 Radish: 0-3 / 0-3 Glass: 0-2 / 0-2	Uniform