Supplementary material: TASKOGRAPHY – Evaluating robot task planning over large 3D scene graphs

Christopher Agia*[1], Krishna Murthy Jatavallabhula*[2], Mohamed Khodeir[1], Ondrej Miksik[3], Vibhav Vineet[3], Mustafa Mukadam[4], Liam Paull[2], and Florian Shkurti[1,5]

[1]University of Toronto, [2]Montreal Robotics and Embodied AI Lab, Mila, Université de Montréal, [3]Microsoft, [4]Facebook AI Research, [5]Vector Institute

Abstract: This supplementary material discusses additional details and design choices for the TASKOGRAPHY benchmark. We also provide results on additional domains, and discuss SCRUB and its favourable properties in greater detail.

Please visit our project page for more details, including a video abstract.

Keywords: Robot task planning, 3D scene graphs, learning to plan, benchmarks

1 Benchmark Details

The TASKOGRAPHY benchmark comprises 20 robot task planning domains over 3D scene graphs (3DSGs). In the main paper, we detailed the *Rearrangement(k)*, *Courier(n,k)*, *Lifted Rearrangement(k)*, and *Lifted Courier(n, k)* task definitions following the recently proposed Rearrangement challenge [1]. Table. 1 lists the set of *lifted* objects in each planning domain. In all problems, we have one instance of an *agent*, but several ground objects corresponding to the other categories.

1.1 TASKOGRAPHY domain construction: Parsing Gibson 3DSGs

We parse the 3DSGs created over Gibson [2, 3] mapping scene entities to objects and structural relations to predicates over objects. We retain key connectivity constraints that govern traversable paths between locations in the same place, places in the same room, and between rooms. Because room connectivity data not is provided in the original database, we estimate it by computing a minimal spanning tree over rooms in the 3DSGs with edge weights reflecting the Euclidean distance between room centroids. For larger scenes, we impose a single connection between rooms in different floors (e.g., one set of stairs). Several additional properties are used to express the state of agent and interactable objects, and to associate each of them to a particular location in the 3DSG.

Table 1: Evaluated 3DSG planning domains in TASKOGRAPHY and object types present in each. Domains are further partitioned into tiny and medium splits akin to the 3DSGs provided over Gibson [2, 3]. Scene entities are instantiated as a particular object type according to their semantic class.

	n	k	Agent	Room	Place	Location	Receptacle	Item	Bagslot	Receptacle Class	Item Class
Rearr(k)	-	{1, 2, 5, 10}	✓	✓	✓	✓	✓	✓	✗	✗	✗
Cour(n, k)	{3, 5, 7, 10}	{5, 10}	✓	✓	✓	✓	✓	✓	✓	✗	✗
Lifted Rearr(k)	-	{5}	✓	✓	✓	✓	✓	✓	✗	✓	✓
Lifted Cour(n, k)	{5}	{5}	✓	✓	✓	✓	✓	✓	✓	✓	✓

An assignment of values to all possible properties over objects defines a symbolic *state* in the planning problem; hence, actions taken by the robot in TASKOGRAPHY alter the symbolic state of the 3DSG. We observe a significant variation in the size of the state space between different types of domains as a result of the varying subsets of object and predicate types used to express their respective tasks (see Table. 2). For instance, the *Rearrangement(k)* task represents the lowest complexity domain on TASKOGRAPHY and is thereby defined by the smallest subset of object types, predicates, and actions available to the robot. In contrast, the *Lifted Courier(n, k)* extends the *Rearrangement(k)* task definition with bagslots enabling stow and retrieve operators, as well as receptacle classes and item classes to express lifted class relations in the 3DSG at particular state.

*Authors contributed equally. Order determined by academic juniority.

We leverage **task samplers** built into TASKOGRAPHY-API for generating large-scale and diverse datasets of planning problems over 3DSGs. In a two step process the task samplers automatically parse 3DSGs into plannable symbolic representations (i.e., embedding the agent forms the initial state $\mathcal{I}$) before composing goal literals over randomly sampled scene entities. For grounded problems, goals are conjunctions of inReceptacle literals expressed over randomly sampled item and receptacle target ground instances. For lifted problems, goal are conjunctions of classRelation literals expressed over randomly sampled item and receptacle target class relations.

Table 2: Structural relations of 3DSGs and the state of the robot and interactable objects (i.e., items and receptacles) are captured with an assignment of values to all possible predicates over objects. The most challenging *Lifted Courier(n, k)* is the only domain to incorporate all relations, while other domain types in TASKOGRAPHY require only a subset of the properties and relations.

Object (:types)	**Agent**	**Room**	**Place**	**Location**	**Receptacle**	**Item**	**Bagslot**	**Receptacle Class**	**Item Class**
Agent	holdsAny	inRoom	inPlace	atLoc	-	holdsItem	-	-	-
Room	inRoom	connected	placeInRoom + roomCenter	-	-	-	-	-	-
Place	inPlace	placeInRoom + roomCenter	-	locInPlace + placeCenter	-	-	-	-	-
Location	atLoc	-	locInPlace + placeCenter	-	recepAtLoc	itemAtLoc	-	-	-
Receptacle	-	-	-	recepAtLoc	recepOpened	inRecep	-	recepClass	-
Item	holdsItem	-	-	itemAtLoc	inRecep	small + medium + large	inSlot	-	itemClass
Bagslot	-	-	-	-	-	inSlot	slotHoldsAny	-	-
Receptacle Class	-	-	-	-	recepClass	-	-	-	classRelation
Item Class	-	-	-	-	-	itemClass	-	classRelation	-

1.2 Domain specifications

To provide further clarity on the four task categories (*Rearrangement(k)*, *Courier(n,k)*, *Lifted Rearrangement(k)*, and *Lifted Courier(n, k)*) from which our 3DSG planning domains are constructed, we herein outline hypothetical problem instances involving but a fraction of the objects, attributes, and relations available in TASKOGRAPHY. Let the environment consist of *v* `rooms` connected by *e* undirected traversability constraints; e.g., connected(roomA, roomB). The spatial hierarchy of 3DSGs [2, 4] is induced by the appropriate application of structural relations (see Table. 2) to a discrete set of `places` in each `room`, and `locations` in each `place`; e.g., placeInRoom(placeD, roomC), locInPlace(locF, placeD). The lowest level of the spatial hierarchy (`locations`) encodes all occupiable positions for the `agent`, `items`, and `receptacles` in the scene; e.g., atLoc(agent, locationB), itemAtLoc(mugA, locationD), recepAtLoc(fridgeC, locationG). Such relations equate to logical predicates in [5] and can be altered by the agent should the required preconditions of an action be met in the current state; e.g., ¬holdsAny(agent) and ∧(atLoc(agent, locX), itemAtLoc(mugA, locX)) are preconditions for PICKUPITEM(mugA, agent).

As mentioned in Sec. 1.1, the goals in grounded planning problems are specified with inReceptacle literals. Concretely, a *Rearrangement(k)* task for $k = 1$ requires the agent to pick-and-place a ground item in a ground receptacle, where each object in the goal is uniquely identified; e.g., $G =$ inReceptacle(mugA, fridgeC). By extension, a *Rearrangement(k)* task for $k = 2$ is solved *iff* the agent derives a state satisfying the conjunction of two inReceptacle goal literals; e.g., $G = \wedge$(inReceptacle(mugA, fridgeC), inReceptacle(plateD, shelfB)). The *Courier(n, k)* domains attribute weights ($w \in 1, 2, 3$ units) to items based on their volume, and equips the agent with a knapsack of fixed capacity *n* to stow and retrieve items as it traverses the scene. While the knapsack in *Courier(n, k)* enables planners to exploit stowing capacity to compute lower cost solutions (at the expense of task complexity) in comparison to *Rearrangement(k)*, goals are identically specified between the two task categories since they are both considered grounded.

In stark constrast, lifted planning problems are specified with classRelation literals expressed over item-receptacle class combinations. For instance, the following *Lifted Rearrangement(k)* or *Lifted Courier(n, k)* domain with $k = 2$, $G = \wedge$(classRelation(cup, cupboard), classRelation(plate, sink)), requires the agent to place **at least one** cup in a cupboard and plate in a sink for the task to be complete. This disambiguates the planner which is no longer able to exploit ground objects featured in the goal as heuristic *landmarks*, and reduces the effectiveness of deterministic graph sparsification techniques such as SCRUB. As in the grounded domain variants, the goal specifications for both the *Lifted Rearrangement(k)* and *Lifted Courier(n, k)* are identical.

1.3 Symbolic environment interaction

The **action space** of the most complex domain in TASKOGRAPHY equips the agent with 16 operators where only a subset are feasible at any given state. Below, we describe but a few of these operators which demonstrate motion through 3DSG hierachies and object-level robot interaction.

- GOTOROOM: The robot moves from the door of its current room to the door of the target room if the rooms are *connected*.
- GOTOPLACE: The robot moves from the center of its current place to the center of the target place if the places are in the same room.
- GOTOLOCATION: The robot moves from the current location to the target location if the locations are in the same place.
- OPENRECEPTACLE: The robot opens a queried *openable* receptacle.
- CLOSERECEPTACLE: The robot closes a queried *openable* receptacle.
- PICKUPITEM: The robot picks-up an item at a particular location with a free gripper; three operator variations for picking from non existent, non-opening, and opening receptacles.
- PLACEITEM: The robot places an in-gripper item at a particular location; two operator variations for placing in non-opening and opening receptacles.
- STOWITEM: The robot stows an in-gripper item in its knapsack: three operator variations for small, medium, and large items consuming increasing numbers of bagslots.
- RETRIEVEITEM: The robot retrieves an item from its knapsack into its gripper; three operator variations for small, medium, and large items freeing increasing number of bagslots.

Should the preconditions for any of these actions not be satisfied, the action is deemed invalid.

2 SCRUB: Discussion and analysis

In the main paper, for sake of brevity, we only discussed the applicability of SCRUB to grounded planning problems with deterministic transitions. However, by design, SCRUB may be applied to any planning problem: *lifted* or *grounded*, with *deterministic* or *stochastic* transitions.

In *lifted* planning problems, we modify SCRUB to trivially include all ground object tuples that satisfy goal conditions into the initial sufficient object set. This in-turn ensures that all of these ground objects are reachable from the start state, ensuring a satisficing plan exists. However, this conservative strategy may resulting in retaining more objects than minimally required – this is where SEEK can be applied to opportunistically retain important objects instead.

In a similar vein, for *stochastic* transitions, we modify SCRUB to include all binary predicates resulting from all possible stochastic transitions from a given node.

We now prove that SCRUB results in a minimal scene subgraph for all grounded planning problems.

Proposition 1. SCRUB *is complete and results in a minimal scene subgraph for all grounded planning problems over the scenegraph domain.*

Proof. We prove the minimality of SCRUB by demonstrating that whenever we prune a node from a SCRUBBED scenegraph, the resultant planning problem is unsolvable. Assume that we prune a node n from a SCRUBBED 3DSG $\hat{G}$. Recall the types of nodes we have in the 3DSG: `agent, room, place, receptacle, item, floor, building`.

1. If n is of type `agent` or `building`, the problem is unsolvable, by construction.

2. If n is of type `item`, removing it would render the goal state unreachable — recall that $\hat{G}$ only retains `item` nodes that feature in the goal state.

3. If n is of type `receptacle`, it is retained in $\hat{G}$ either because (a) it is required to access a goal object of type `item`, or (b) it is a goal `receptacle` (i.e., a target location an `item` must be moved into). Removing n will thus render one of the objects in the goal state unreachable.

4. If n is of type `place`, `room` or `floor`, $n \in \hat{G}$ because n directly features in the goal state, or because n is required to traverse from the start state to the goal state (e.g., rooms that connect the start and goal rooms, etc.).

Since pruning any of these nodes renders the problem unsolvable, the SCRUBBED graph $\hat{G}$ is a minimal scene subgraph for the grounded planning problem considered. □

3 Additional results on TASKOGRAPHY domains

In this section, we provide results over several extended domains from the TASKOGRAPHY benchmark. Please see Tables 3, 4, 5, 6, 7, 8.

Table 3: Performance of planners over the *Rearrangement(k)*-Tiny tasks. For all metrics, lower values indicate better performance.

		Rearr(1) Tiny			**Rearr(2)** Tiny			**Rearr(10)** Tiny		
	Planner	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**
optimal	**FD-seq-opt-lmcut**	15.77	24.81	0.04	**25.80**	104.47	0.55	-	-	1.00
	SatPlan	14.77	10.35	0.45	26.67	3.27	0.67	-	-	1.00
	Delfi	15.13	0.36	0.16	29.10	27.77	0.29	-	-	1.00
	DecStar-opt-fb	-	-	1.00	-	-	1.00	-	-	1.00
	MCTS	-	-	1.00	-	-	1.00	-	-	1.00
satisficing	**FF**	16.71	0.19	**0.00**	34.44	0.55	**0.00**	162.61	7.04	0.07
	FF-X	16.71	0.25	**0.00**	34.44	0.58	**0.00**	162.30	7.39	0.09
	FD-lama-first	15.19	2.96	0.33	38.47	3.25	0.18	205.89	7.68	0.51
	Cerberus-sat	**11.50**	12.00	0.85	-	-	1.00	-	-	1.00
	Cerberus-agl	14.77	5.13	0.45	33.00	7.30	0.49	186.07	9.04	0.73
	DecStar-agl-fb	14.72	2.62	0.55	34.96	2.58	0.58	193.00	6.78	0.85
	BFWS	15.56	0.90	0.22	32.16	0.37	0.18	**160.93**	0.57	0.18
	Regression-plan	-	-	1.00	-	-	1.00	-	-	1.00
learn	**Relational policy [6]**	-	-	1.00	-	-	1.00	-	-	1.00
	PLOI [7]	16.45	**0.00***	**0.00**	37.04	**0.00***	**0.00**	221.71	**0.18**	**0.00**

Table 4: Performance of planners over the *Rearrangement(k)*-Medium tasks. For all metrics, lower values indicate better performance.

		Rearr(1) Medium			**Rearr(2)** Medium			**Rearr(10)** Medium		
	Planner	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**
optimal	**FD-seq-opt-lmcut**	15.53	19.68	0.06	**27.13**	125.69	0.41	-	-	1.00
	SatPlan	14.98	11.98	0.33	28.23	5.45	0.50	-	-	1.00
	Delfi	15.40	3.62	0.16	29.13	12.79	0.28	-	-	1.00
	DecStar-opt-fb	15.42	41.35	0.93	28.50	111.53	0.91	-	-	1.00
	MCTS	-	-	1.00	-	-	1.00	-	-	1.00
satisficing	**FF**	16.45	0.25	**0.00**	32.87	0.41	**0.00**	159.04	5.30	0.09
	FF-X	16.45	0.21	**0.00**	32.87	0.45	**0.00**	159.80	5.02	0.08
	FD-lama-first	15.51	2.48	0.21	39.20	2.77	0.20	208.28	6.35	0.49
	Cerberus-sat	**11.20**	10.17	0.88	-	-	1.00	-	-	1.00
	Cerberus-agl	15.18	6.10	0.34	32.20	6.40	0.33	176.60	8.91	0.72
	DecStar-agl-fb	15.36	2.15	0.58	36.35	2.40	0.59	211.16	7.20	0.82
	BFWS	15.42	0.60	0.23	30.65	0.44	0.27	**151.17**	0.41	0.23
	Regression-plan	-	-	1.00	-	-	1.00	-	-	1.00
learn	**Relational policy [6]**	-	-	1.00	-	-	1.00	-	-	1.00
	PLOI [7]	16.44	**0.00***	**0.00**	36.19	**0.00***	**0.00**	213.43	**0.17**	**0.00**

Table 5: Performance of planners over the *Courier(n, k)*-Tiny tasks. For all metrics, lower values indicate better performance.

		Cour(3, 10) Tiny			**Cour(5, 10)** Tiny			**Cour(7, 10)** Tiny			**Cour(10, 10)** Tiny		
	Planner	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**
satisficing	**FF**	146.35	7.57	0.13	136.38	7.97	0.33	127.88	6.84	0.55	124.93	14.62	0.73
	FF-X	144.80	8.34	0.11	137.05	7.49	0.31	128.42	8.34	0.53	126.31	15.21	0.71
	FD-lama-first	175.15	8.31	0.53	159.64	7.31	0.55	156.12	6.97	0.55	145.00	7.50	0.56
	Cerberus-sat	-	-	1.00	-	-	1.00	-	-	1.00	-	-	1.00
	Cerberus-agl	**137.87**	10.79	0.73	127.30	17.61	0.82	138.25	21.65	0.93	-	-	1.00
	DecStar-agl-fb	140.47	4.52	0.69	**124.62**	4.65	0.71	**120.20**	4.04	0.73	**117.73**	6.98	0.73
	BFWS	160.18	1.19	0.18	159.17	0.94	0.25	159.90	1.80	0.29	153.93	4.28	0.45
	Regression-plan	-	-	1.00	-	-	1.00	-	-	1.00	-	-	1.00
learn	**Relational policy [6]**	-	-	1.00	-	-	1.00	-	-	1.00	-	-	1.00
	PLOI [7]	193.55	**0.22**	**0.00**	179.36	**0.26**	**0.00**	172.87	**0.37**	**0.00**	167.38	**0.71**	**0.00**

Table 6: Performance of planners over the *Courier(n, k)*-Medium tasks. For all metrics, lower values indicate better performance.

		Cour(3, 10) Medium			**Cour(5, 10)** Medium			**Cour(7, 10)** Medium			**Cour(10, 10)** Medium		
	Planner	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**
satisficing	**FF**	**141.89**	4.94	0.07	133.46	6.29	0.20	128.41	6.62	0.24	117.50	14.27	0.78
	FF-X	**141.89**	4.47	0.07	133.50	5.80	0.19	128.19	6.72	0.24	118.67	15.52	0.77
	FD-lama-first	180.38	7.11	0.40	166.35	6.27	0.45	156.34	4.92	0.29	141.75	6.80	0.63
	Cerberus-sat	-	-	1.00	-	-	1.00	-	-	1.00	-	-	1.00
	Cerberus-agl	148.41	10.17	0.74	**133.31**	11.50	0.77	**125.73**	12.99	0.83	**109.56**	15.58	0.95
	DecStar-agl-fb	154.07	6.45	0.66	142.42	4.01	0.61	132.60	4.50	0.58	128.58	7.60	0.70
	BFWS	151.09	0.60	0.27	152.61	0.66	0.20	152.71	1.13	0.21	153.02	2.81	0.30
	Regression-plan	-	-	1.00	-	-	1.00	-	-	1.00	-	-	1.00
learn	**Relational policy [6]**	-	-	1.00	-	-	1.00	-	-	1.00	-	-	1.00
	PLOI [7]	182.31	**0.20**	**0.00**	169.20	**0.24**	**0.00**	161.90	**0.34**	**0.00**	152.19	**0.61**	**0.00**

Table 7: Performance of planners over the *Lifted Rearrangement(k)* domains. For all metrics, lower values indicate better performance.

		Lifted Rearr(5, 5) Tiny			**Lifted Rearr(5, 5)** Medium		
	Planner	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**
satisficing	**FF**	62.86	3.40	0.47	61.90	3.04	0.37
	FF-X	67.88	3.48	0.89	61.78	2.30	0.72
	FD-lama-first	66.81	3.20	0.49	71.15	4.11	0.46
	Cerberus-sat	-	-	1.00	-	-	1.00
	Cerberus-agl	60.50	7.62	0.60	64.26	6.74	0.57
	DecStar-agl-fb	66.30	3.02	0.71	77.00	3.08	0.71
	BFWS	**56.90**	0.94	0.41	**55.36**	0.80	0.43
	Regression-plan	-	-	1.00	-	-	1.00
learn	**Relational policy [6]**	-	-	1.00	-	-	1.00
	PLOI [7]	78.68	**0.22**	**0.24**	76.62	**0.22**	**0.24**

Table 8: Performance of planners over the *Lifted Courier(n, k)* domains. For all metrics, lower values indicate better performance.

		Lifted Cour(5, 5) Tiny			**Lifted Cour(5, 5)** Medium		
	Planner	**Len.**	**Time**	**Fail**	**Len.**	**Time**	**Fail**
satisficing	**FF**	**57.74**	4.03	0.44	57.38	4.81	0.37
	FF-X	61.19	7.56	0.77	60.05	3.79	0.64
	FD-lama-first	61.13	3.34	0.56	63.19	3.31	0.45
	Cerberus-sat	-	-	1.00	-	-	1.00
	Cerberus-agl	59.19	7.05	0.77	59.61	7.45	0.68
	DecStar-agl-fb	58.75	4.46	0.71	63.93	3.85	0.68
	BFWS	61.92	2.30	0.43	**56.14**	0.67	0.38
	Regression-plan	-	-	1.00	-	-	1.00
learn	**Relational policy [6]**	-	-	1.00	-	-	1.00
	PLOI [7]	71.71	**0.26**	**0.26**	69.92	**0.46**	**0.30**

References

[1] D. Batra, A. X. Chang, S. Chernova, A. J. Davison, J. Deng, V. Koltun, S. Levine, J. Malik, I. Mordatch, R. Mottaghi, et al. Rearrangement: A challenge for embodied ai. *arXiv preprint arXiv:2011.01975*, 2020.

[2] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2019.

[3] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2018.

[4] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone. 3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans. In *Robotics Science and Systems (R-SS)*, 2020.

[5] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins. Pddl-the planning domain definition language. 1998.

[6] O. Rivlin, T. Hazan, and E. Karpas. Generalized planning with deep reinforcement learning. *arXiv preprint arXiv:2005.02305*, 2020.

[7] T. Silver, R. Chitnis, A. Curtis, J. Tenenbaum, T. Lozano-Perez, and L. P. Kaelbling. Planning with learned object importance in large problem instances using graph neural networks. *AAAI International conference on Artificial Intelligence*, 2020.