Lazy-DaSH: Lazy Approach for Hypergraph-based Multi-robot Task and Motion Planning

Seongwon Lee[1], James Motes[1], Isaac Ngui[1], Marco Morales[2], and Nancy M. Amato[1]

1 Parasol Lab, University of Illinois at Urbana-Champaign

2 Department of Computer Science, Instituto Tecnológico Autónomo de México (ITAM)

Motivation

- Multi-robot task and motion planning (MR-TAMP) are used in many applications
  - Autonomous manufacturing
- The multi-manipulator object rearrangement problem exemplifies the challenges of MR-TAMP
  - Task plan: A sequence of pick, place, and hand-over
  - Motion plan: A collision-free coordinated motions

Backgrounds

Integrated Task and motion planning [1]

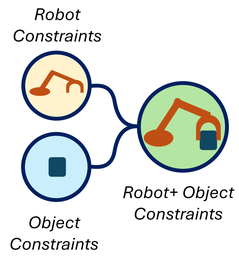
- **Sequencing first**
  - Plan high-level actions before checking the satisfaction of the action's constraints
- **Satisfaction first**
  - Focus on satisfying constraints before creating an action sequence
- **Interleaved**
  - Dynamically balance of both methods

Multi-robot Task and motion Planning

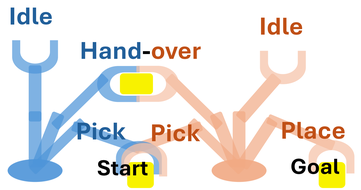
- **Decoupled**
  - Decompose search space into independent robot state spaces
  - Faster planning times but low coordination
- **Coupled**
  - Couples search space to a unified robot state space
  - Slower planning times but high coordination
- **Hybrid**
  - Balance the strengths of both while minimizing their weaknesses

Decomposable State Space Hypergraph (DaSH) [2]

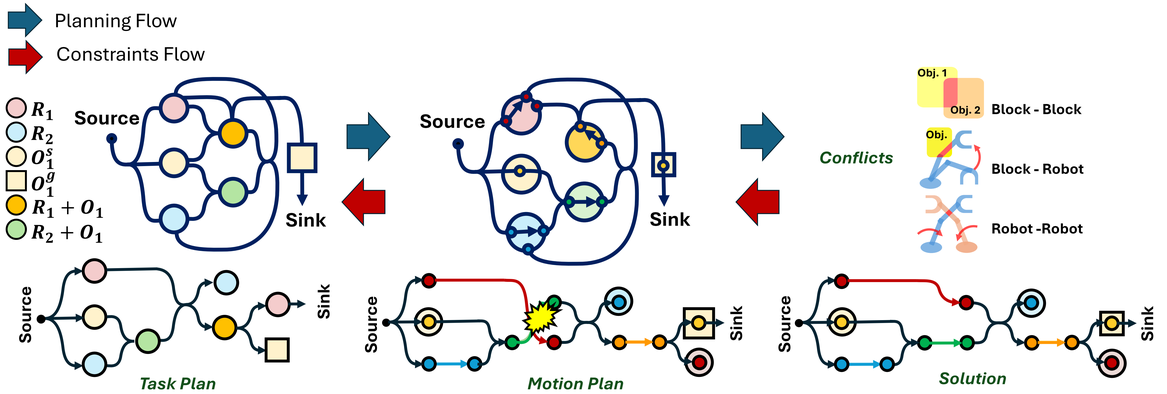
- Represent decomposed robot state space as hypergraphs
- Hybrid approach
  - Focuses effort only where coordination is needed
- Interleaved approach
  - Iterates sampling robot constraints and querying the motion plan
- The planning time is still excessive due to
  - Unnecessary motion feasibility check
  - Excessive size of representation to query



Proposed Approach



- ***A hierarchical structure***
  - Uses a queryable task space hypergraph identifying task plans
- ***Lazy motion validation***
  - Assumes all motions are valid during task planning
  - Defers the motion validity checks until the motion query
- **Constraint feedback mechanism**
  - Informs infeasible motions at each layer
  - Generates task/motion constraints



Step1. Task Planning Layer

Task Space Representation

- A high-level hypergraph representing the robot interactions
- Captures pick, place, and hand-over interaction without motion details

Task Query

- "Unfold" the task space hypergraph to find a valid action sequence
- Outputs a task plan without motion validation

Step2. Motion Planning Layer

Motion Representation

- Detailed hypergraph with motions
- Generate *lazy* roadmaps only for the relevant transitions identified by the task plan

Motion Query

- Validate the motion path guided by the task plan
- Failed motion plan creates motion constraints or task constraints

Step3. Conflict Resolution Layer

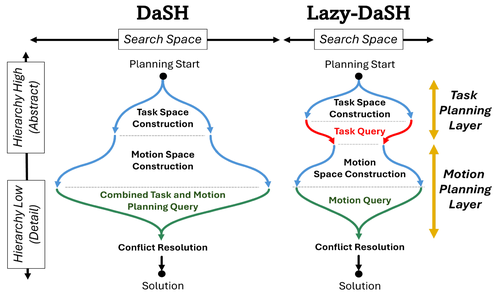
Conflicts

- Defined by a collision
- Block-Block, Block-Robot, and Robot-Robot

Constraints Feedback Mechanism

- Motion constraints provide constraints to the motion planning layer
- Task constraints provide task-level constraints to the task planning layer

Comparison



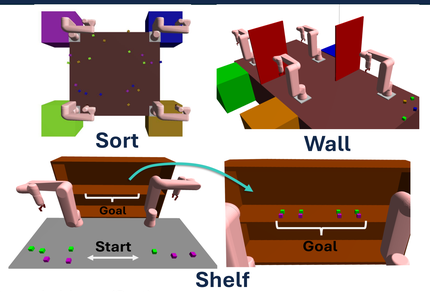
DaSH (Original)

- Focuses on satisfying constraints
- Pre-samples all feasible transitions and motions before querying
- Expands the search space consecutively, adding unnecessary constraints
- Increases time for representation construction and querying

Lazy-DaSH (Proposed)

- Focuses on sequencing
- Identifies minimum constraints needed for task completion
- Reduces exponential growth in representation size

Validation Scenarios



Sort

- Demonstrate the scalability of the algorithm

Wall

- Show the ability to identify infeasible actions with constraints feedback mechanism

Shelf

- Demonstrate the capacity to handle precedence constraints

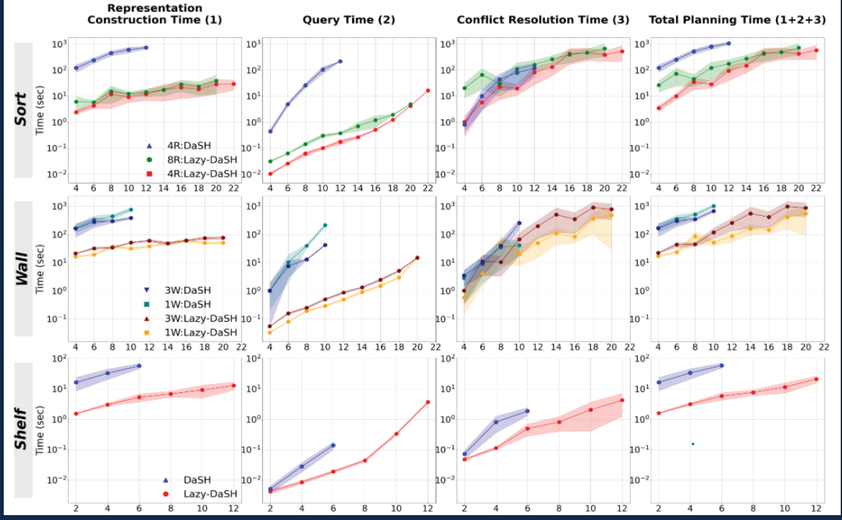
Validation

Scalability

- Lazy-DaSH improves scalability with more robots and objects (Sort)
  - Efficient control of representation size

Planning time

- Representation construction time reduced by up to two orders of magnitude (Sort, Wall, Shelf)
  - The task query phase identifies relevant motion construction
  - Lazy motion validation focuses on validating only relevant motions
- Query time reduced by up to three orders of magnitude (Sort)



ACKNOWLEDGEMENT

This work was supported in part by the U.S. National Science Foundation's "Expeditions: Mind in Vitro: Computing with Living Neurons" under award No. IIS-2123781, and by the IBM-Illinois Discovery Accelerator Institute and the Center for Networked Intelligent Components and Environments (C-NICE) at the University of Illinois.

References

[1] Garrett, Caelan Reed, et al. "Integrated task and motion planning." *Annual review of control, robotics, and autonomous systems* 4.1 (2021): 265-293.

[2] Motes, James, et al. "Hypergraph-based multi-robot task and motion planning." *IEEE Transactions on Robotics* (2023).