Integrating Knowledge Compilation with Reinforcement Learning for Routes

Jiajing Ling, Kushagra Chandak, Akshat Kumar
{jjling.2018, kushagrac, akshatkumar}@smu.edu.sg
SMU SINGAPORE MANAGEMENT UNIVERSITY
School of Computing and Information Systems

Motivation

- **We are surrounded by autonomous systems**
- **Autonomous vehicles need to satisfy some constraints**


Intersection constraints


No loop constraints


Patrolling Constraints

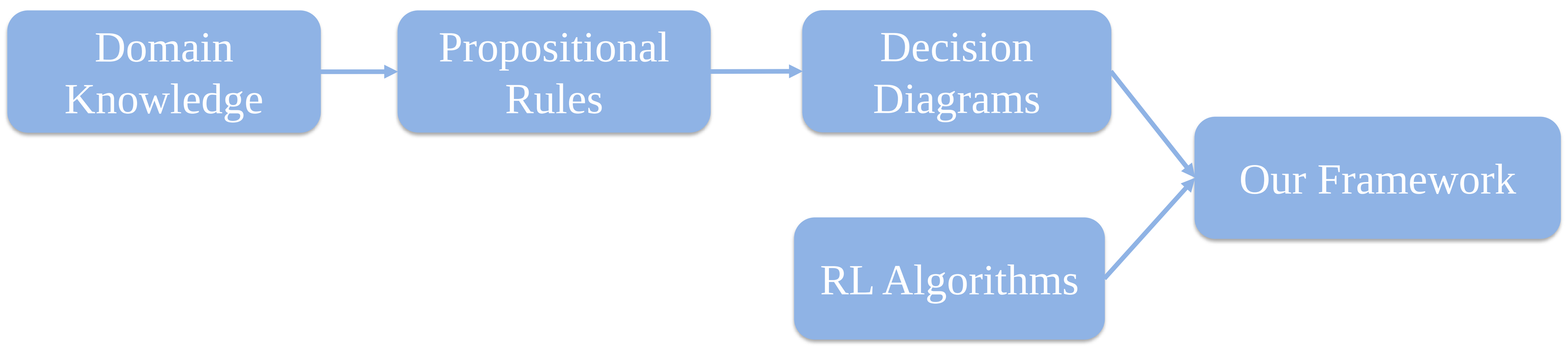

Pick-up and drop-off constraints

- **Training such systems using reinforcement learning is sample inefficient**

How to encode domain constraints with decision making algorithms?

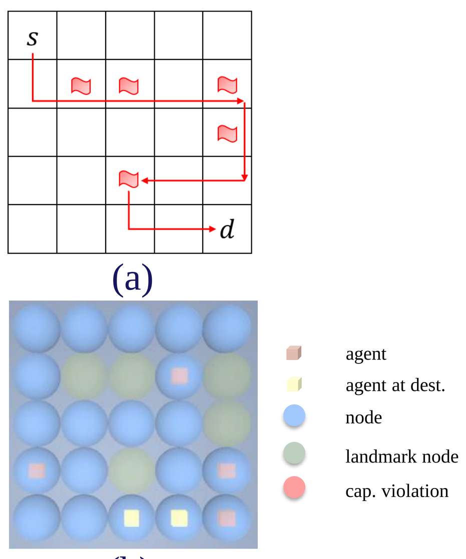
Our Contributions

- **Domain knowledge compilation using Boolean logic based decision diagrams**
- **Integration of domain knowledge with deep RL in a modular fashion**
- **Fast querying of the compiled decision diagrams**



Problem Setting

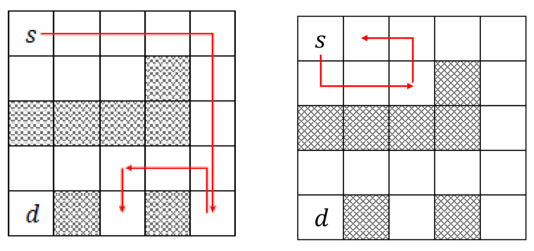
- **MARL for Multiagent Path Finding**
  - Under uncertain travelling time
  - Partial observability
- **Agents move from their respective sources to destinations**
  - Minimize travelling time and avoid collisions
  - **Satisfy some constraints**
- **Challenges**
  - Sample inefficiency: several simulations to find a feasible path.
  - Difficult to encode constraints/domain knowledge with model-free RL
- **Examples**
  - A 5x5 open grid with 5 landmarks (denoted as flags)
  - A screenshot from our simulator (for 6 agents)



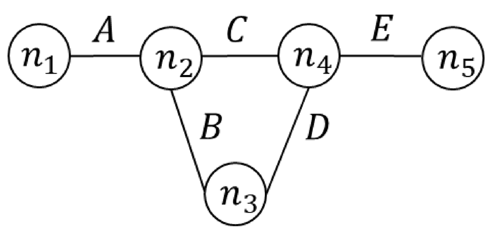
(a)
(b)

Constraints & Boolean Formula

- **Constraints**
  - Simple paths
    - A node that cannot be visited more than once
    - Help avoid deadend and loop
  - Landmarks: visit a set of locations
  - Coverage: visit a set of locations every k steps
- **Propositional rules for constraints**
  - Boolean variable $X_{i,j}$ for edge (i,j)
- **Examples:**
  - Simple path (from n_1 to n_5)
    - $A \wedge \neg B \wedge C \wedge \neg D \wedge E$
  - Landmarks constraint
    - Disjunction of all incident edge variables on a landmark
  - Final constraint: Simple path AND landmarks



deadend loop



Empirical evaluation

- **Number of paths encoded**
  - Can encode a large number of paths
  - SDD nodes is still tractable to represent and reason with
- **Baseline heuristics & Simulation speed**
  - Open grid: shortest path based algo. (GH1)
  - Landmark: max-flow based algo. (GH2)

Size	Open grid	5 Landmarks
10x10	1.08E+13	2.25E+12
20x20	4.59E+51	1.21E+50

Number of encoded paths

Approach	5x5	10x10	20x20
BU-SAT	979.7	38873.6	730031.1
TD-SAT	153.1	1313.0	26915.3
GH1	**6.9**	**42.9**	**473.4**

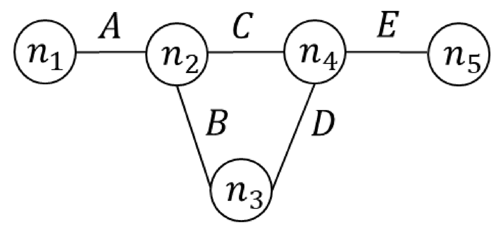
Simulation speed on open grid maps

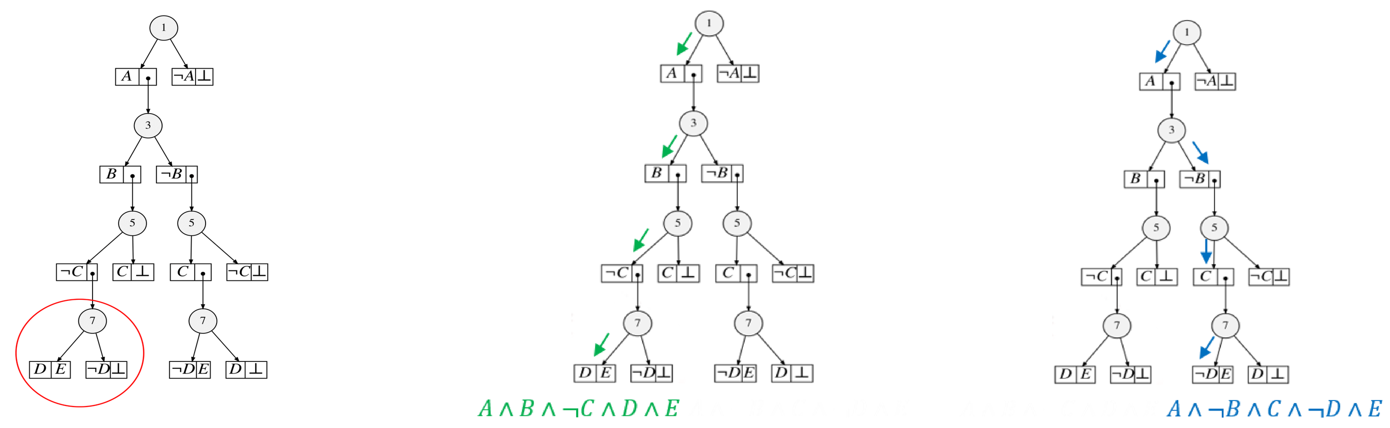
Approach	5x5	10x10	20x20
BU-SAT	4513.6	49135.6	1282995.0
TD-SAT	**201.9**	**2061.9**	**43619.2**
GH2	1461.0	85142.9	494469.9

Simulation speed on maps with 5 landmarks

Compact representation of BF

- **Boolean formula for all simple paths (from n_1 to n_5)**
  - $(A \wedge \neg B \wedge C \wedge \neg D \wedge E) \vee (A \wedge B \wedge \neg C \wedge D \wedge E)$
- **Sentential Decision Diagram (SDD[1]):**
  - Decision nodes $(prime_i, sub_i), i = 1, \dots n$
    - E.g the red circle node in Figure (a)
    - 2 elements: $(D, E), (\neg D, \perp)$
    - Boolean formula: $(D \wedge E) \vee (\neg D \wedge \perp)$
  - Terminal nodes (literals e.g., $A, \neg A, \perp, \top$)
  - Encoded paths in an SDD





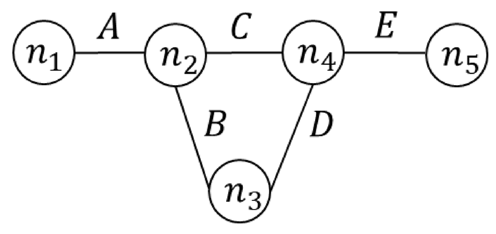
(a) An SDD (b) Encoded path1 (c) Encoded path2

- **Combining constraints**
  - Conjoin the SDDs
  - Modular and fast

[1] SDD: A New Canonical Representation of Knowledge Bases (Darwiche, IJCAI-11)

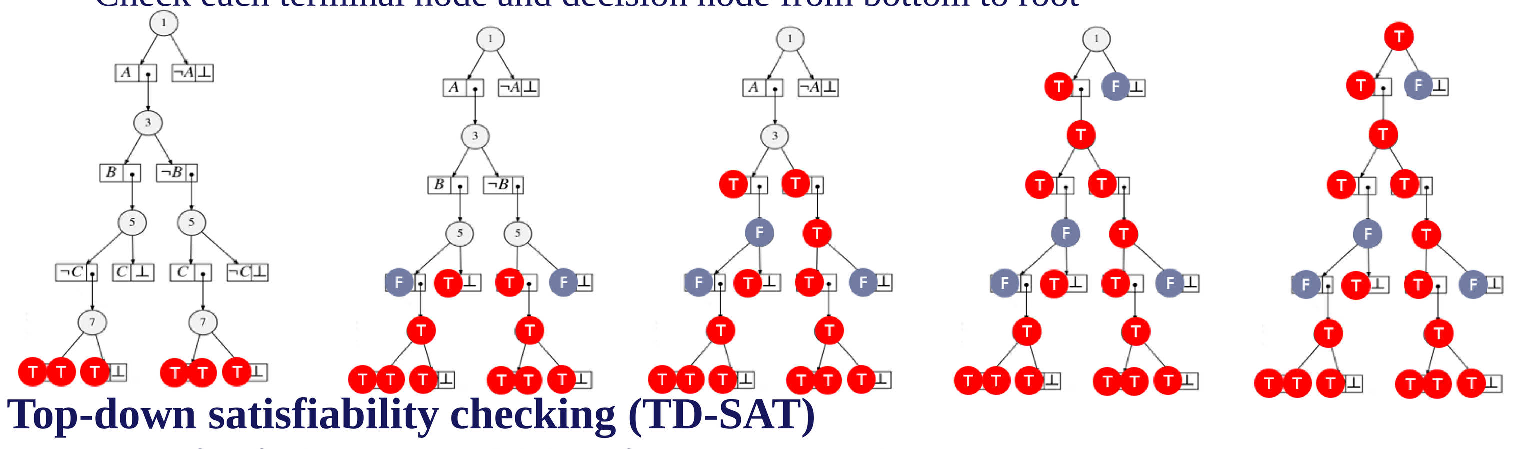
Inferring feasible actions

- **Each step of RL**
  - Find feasible actions given evidence (partial path)
  - Feasible action: An action that satisfies all constraints
- **Example**
  - Given partial path $A \rightarrow C$, is E/D a feasible action?
  - High-level idea
    - E feasible? Given evidence {A,C,E}, check satisfiability of the BF.
    - If satisfiable, then E is a feasible action.
- **Incorporating** feasible actions **in Deep RL**
  - PG: Normalize the last layer of NN over feasible actions set
  - Q-learning: Maximize the Q function only over feasible actions set

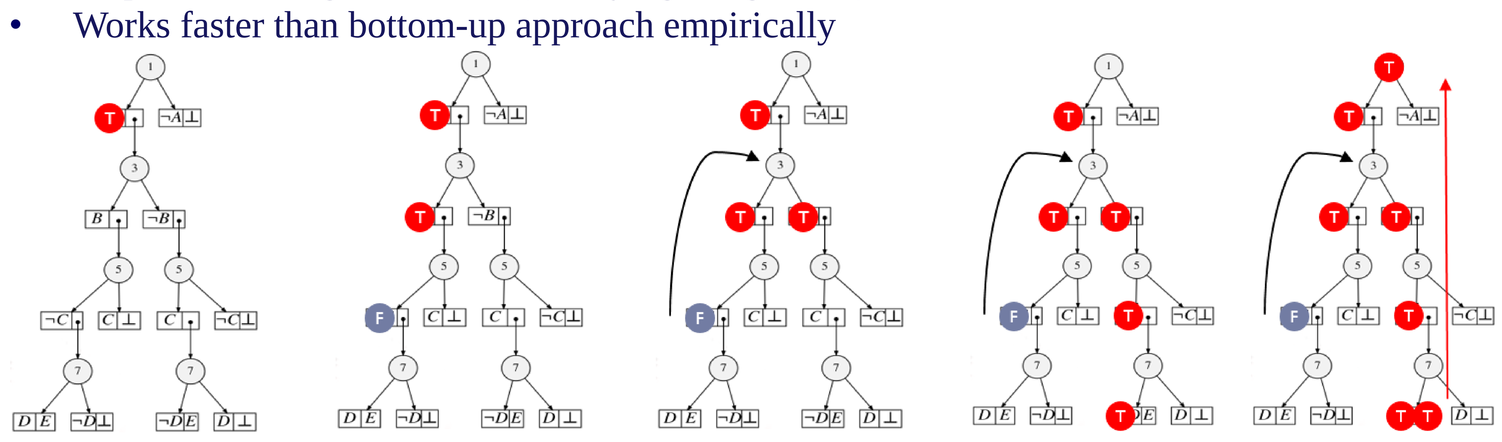


Inference algorithms

- **Given partial path $A \rightarrow C$, is E a feasible action?**
- **Bottom-up satisfiability checking (BU-SAT)**
  - Based on SDD model counting
  - Check each terminal node and decision node from bottom to root

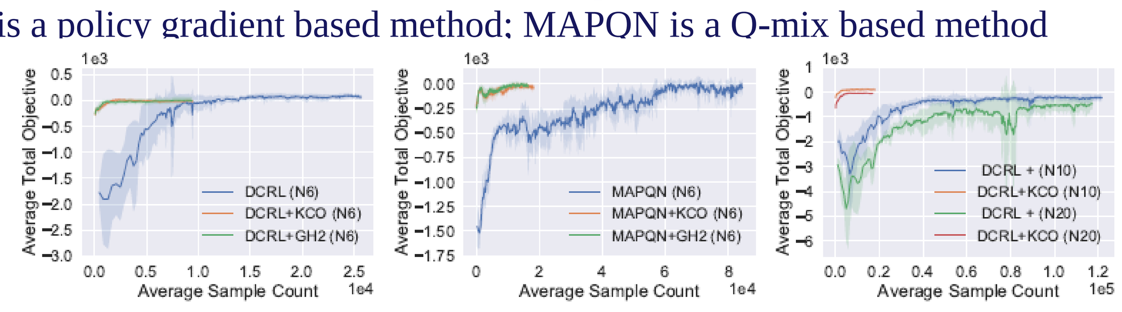


- **Top-down satisfiability checking (TD-SAT)**
  - Stop after finding one model (satisfying assignment)
  - Works faster than bottom-up approach empirically



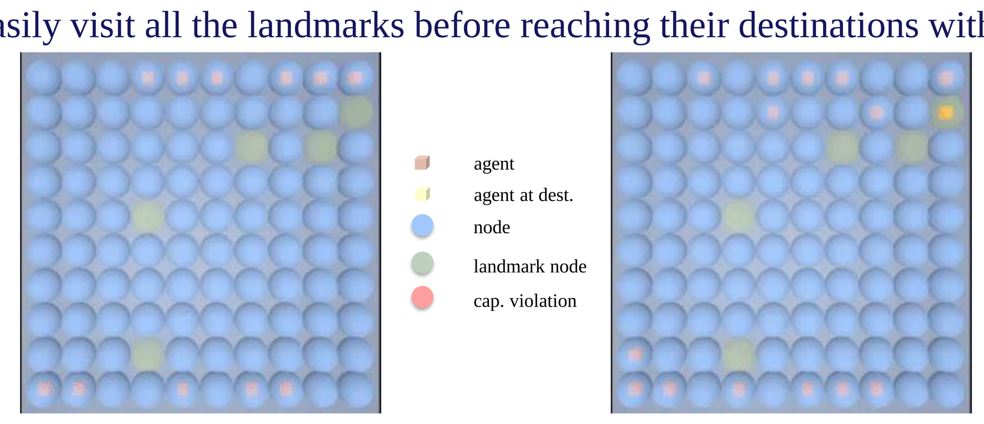
Sample efficiency & solution quality

- **Sample efficiency on maps with 5 landmarks**
  - Results on 5x5 and 10x10 with different number of agents (denoted by N)
  - Combined our framework with different RL algo.
  - DCRL is a policy gradient based method; MAPQN is a Q-mix based method



(a) 5x5 L5 (DCRL) (b) 5x5 L5 (MAPQN) (c) 10x10 L5 (DCRL)

- **Animations from our simulator on instance 10x10, 5 landmark, 20 agents**
  - We show initial 10 training episodes and last 10 training episodes
  - Agents need a lot of exploration to find a feasible path without our framework.
  - Agents can easily visit all the landmarks before reaching their destinations with our framework



DCRL+ KCO DCRL