Towards Specification Learning from Demonstration

Ankit Shah, Julie Shah
{ajshah, arnoldj}@mit.edu

Motivation

- Human apprentices identify whether a given task execution is correct well before expertise in performing the task.
- Imitation learning frames LfD as an inverse reinforcement learning problem. The task specification is implicitly expressed through a reward function or a policy.
- Temporal logics provide a flexible language for expressing specifications for:
  - Synthesis of verifiable controllers[1].
  - Reward signals for reinforcement learning[2].
  - Goal description in domain independent planning[3].
- **Aim:** Learn task specifications from demonstrations. This provides the explicit notion of acceptability of a task execution.
- **Approach**: Bayesian specification inference, a probabilistic model for inferring the temporal task structure as a LTL specification.
- Probabilistic programming languages[4] coupled with appropriately designed prior and likelihood distributions make inference tractable

Bayesian Formulation

$$P(h|\boldsymbol{D}) = \frac{P(h)P(\boldsymbol{D}|h)}{\sum_{h\in \boldsymbol{H}} P(h)P(\boldsymbol{D}|h)}$$

- $\boldsymbol{H} = \boldsymbol{\varphi} = \{\varphi\}$ is the hypothesis space, $P(h)$ must have positive support over all relevant formulas.
- $P(\boldsymbol{D}|h)$ is the likelihood function:
  - Large likelihood value for a demonstration satisfying a complex formula
  - Small likelihood value for a demonstration satisfying a simple formula
  - Number of conjunctions a measure of formula complexity
- We employ probabilistic programming languages to compute posterior using MCMC sampling.

Likelihood Function

Complexity-based likelihood function (CB):

$$\frac{P([\boldsymbol{\alpha}] \vDash \varphi_1|\varphi_1)}{P([\boldsymbol{\alpha}] \vDash \varphi_2|\varphi_2)} = \frac{2^{N_{conj_1}}}{2^{N_{conj_2}}}$$

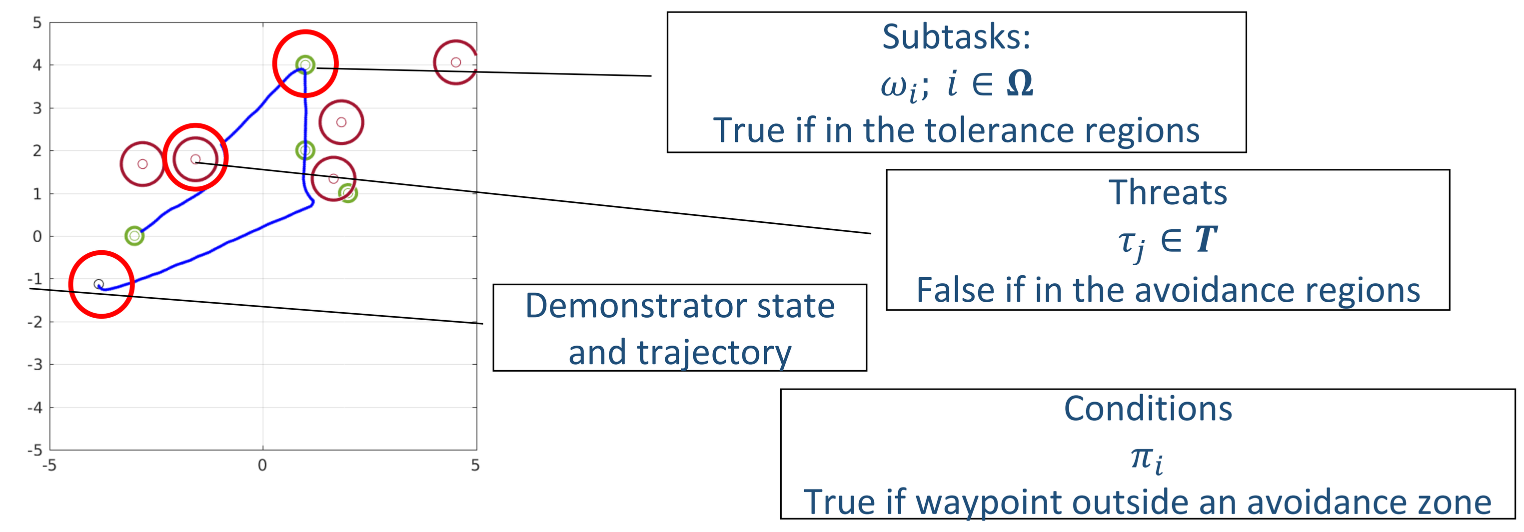
$$\frac{P([\boldsymbol{\alpha}] \neg\vDash \varphi_1|\varphi_1)}{P([\boldsymbol{\alpha}] \vDash \varphi_1|\varphi_1)} = \frac{\epsilon}{2^{N_{conj_1}}}$$

Complexity-independent likelihood function (CI):

$$\frac{P([\boldsymbol{x}] \vDash \varphi|\varphi)}{P([\boldsymbol{x}] \neg\vDash \varphi|\varphi)} = M$$

M is a large number

Approach



- Recursively defined LTL grammar results in an intractable hypothesis space of candidate formulas.
- People commonly use a compositions of temporal behavior templates to express complex specifications[5].

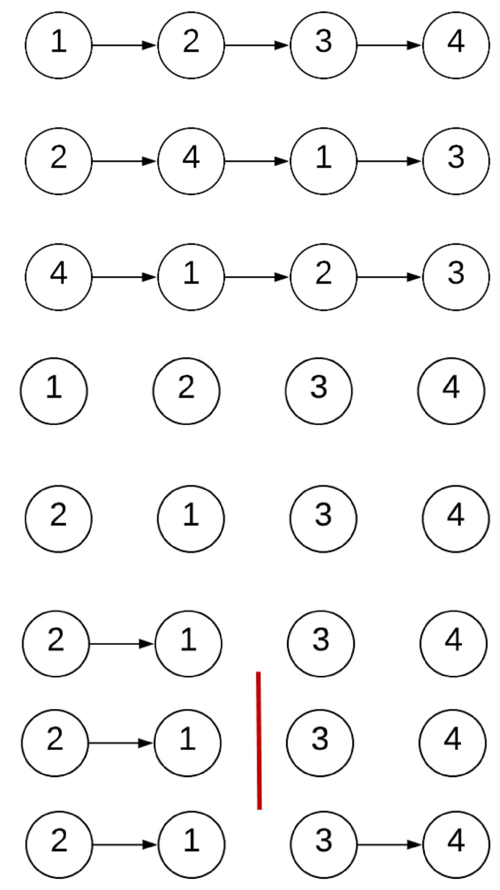
Key Idea: We define prior and likelihood distributions over a relevant part of the LTL formulas based on three templates.

$$\varphi = \varphi_{global} \wedge \varphi_{eventual} \wedge \varphi_{order}$$

$$\varphi = \bigwedge_{\tau\in\boldsymbol{\tau}} \boldsymbol{G}\tau \wedge \bigwedge_{i\,\in\,\boldsymbol{W_1}} (\pi_i \rightarrow \boldsymbol{F}\omega_i) \wedge \bigwedge_{w_1,w_2\in \boldsymbol{W_2}} \pi_i \rightarrow (\neg\omega_{w_2}\boldsymbol{U}\omega_{w_1})$$

- Global satisfaction
- Eventual completion of sub-tasks
- Temporal constraints among sub-tasks

Priors



Linear Chains

Equivalent to a random permutation
`RandomPermutation`$(\boldsymbol{\Omega})$
Repeat n times:
1. Sample a task without replacement

Set of Linear Chains

Multiple disjoint sets - Each a linear chain
`SetOfLinearChains`(Ω, p_{part}):
1. `RandomPermutation`$(\boldsymbol{\Omega})$
2. Repeat $n-1$ times:
   i. `Bernoulli`(p_{part})
   ii. New chain if true, add to chain if false

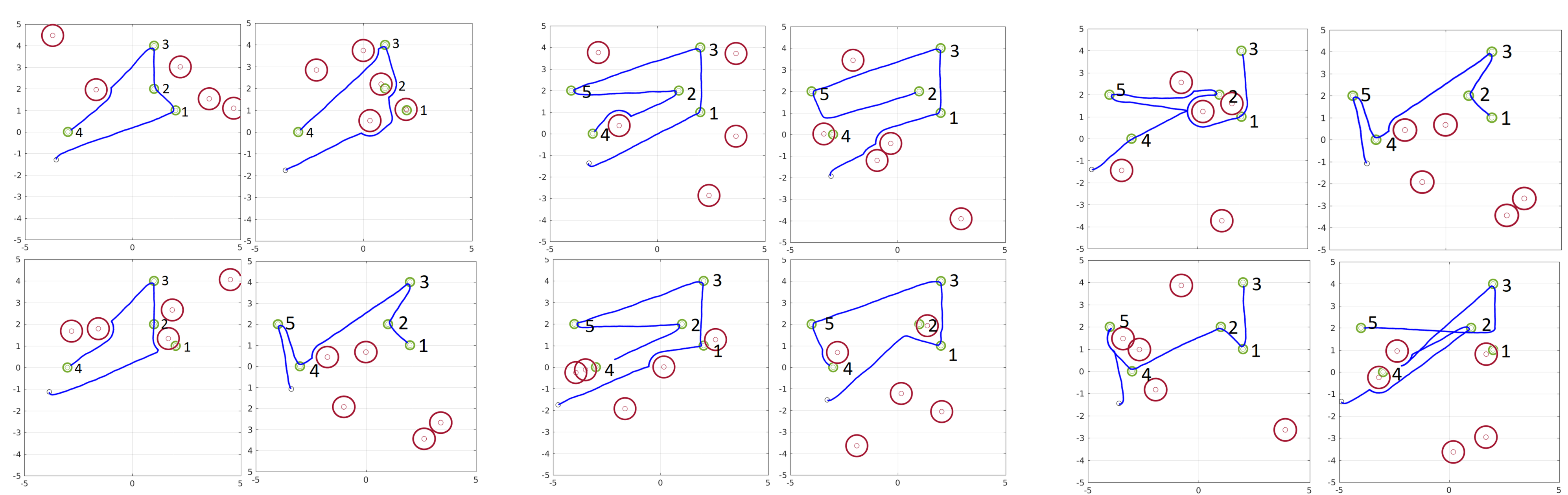
Subset Sampling

`SampleSubset`(A,p)
$A = \{i\}$
Bernoulli trial with success probability p for all elements of A

Prior	φ_{global}	$\varphi_{eventually}$	φ_{order}	Hyperparameters
Prior 1	`SampleSubset`$(\mathbf{T},p_g)$	`SampleSubset`$(\boldsymbol{\Omega},p_e)$	`RandomPermutation`$(\boldsymbol{\Omega})$	p_g, p_e
Prior 2	`SampleSubset`$(\mathbf{T},p_g)$	`SampleSubset`$(\boldsymbol{\Omega},p_e)$	`SetOfLinearChains`$(\boldsymbol{\Omega},p_{part})$	p_g, p_e, p_{part}

Results

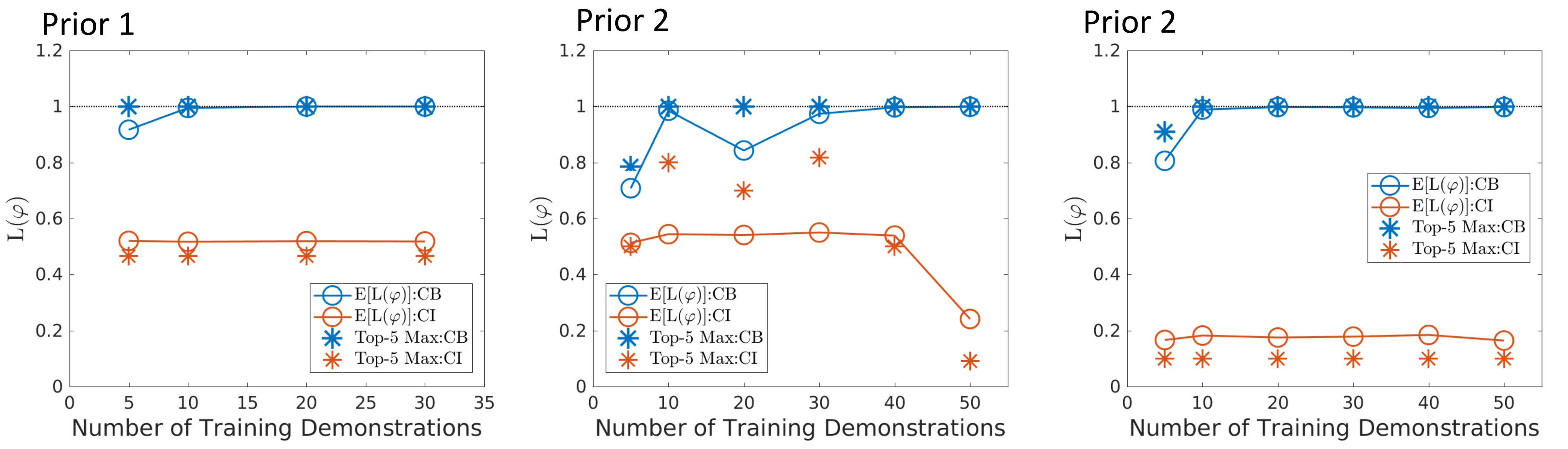
Scenario 1 **Scenario 2** **Scenario 1**



Visit POI in order [1,2,3,4,5]
Prior 1

Visit POI in order {[1,3,5],[2],[4]}
Prior 2

No ordering constraints
Prior 2



Discussion

- ~10 demonstrations are required to fully infer the correct specifications.
- Average runtime 10 and 90 minutes for training set sizes of 5 and 50 respectively.
- Performance with fewer demonstrations in training set is dependent on the prior. With larger training sets it converges to the true specification.

Future Work

- Calibrating priors to better align with human mental model of the task.
- Inferring the Boolean propositions directly from trajectory data.
- Developing and end-to-end learning system that is transparent about its operations and can generate verifiably correct plans.

[1] H. Kress-Gazit, G. E. Fainekos and G. J. Pappas, "Temporal-Logic-Based Reactive Mission and Motion Planning," in IEEE Transactions on Robotics, vol. 25, no. 6, Dec. 2009.
[2] X. Li, C. I. Vasile and C. Belta, "Reinforcement learning with temporal logic rewards," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC.
[3] Camacho, A., Triantafillou, E., Muise, C.J., Baier, J.A. and McIlraith, S.A., "Non-Deterministic Planning with Temporally Extended Goals: LTL over Finite and Infinite Traces", AAAI 2017.
[4] Goodman, N.D. and Stuhlmüller, A., 2014. The design and implementation of probabilistic programming languages.
[5]Matthew B. Dwyer, George S. Avrunin, and James C. Corbett. 1999. Patterns in property specifications for finite-state verification. In Proceedings of the 21st international conference on Software engineering (ICSE '99). ACM, New York, NY, USA.

Visit our website for more!
http://interactive.mit.edu