Learning Space Partitions for Path Planning

Triad members (UC Berkeley affiliation unless otherwise mentioned):
Tianjun Zhang, Kevin Yang,
Joseph E. Gonzalez, Dan Klein, Yuandong Tian (Facebook)

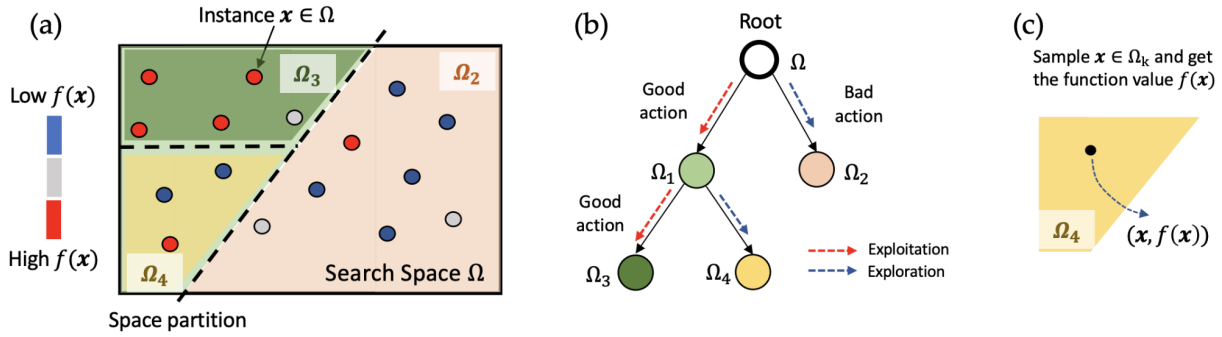

Figure 1: `PlaLaM` extends LaMCTS [44] to path planning. **(a)** Starting from a search space Ω, both `PlaLaM` and LaMCTS first draw a few samples $\mathbf{x} \in \Omega$, then learn to partition Ω into a sub-region Ω_1 with good samples (high $f(\mathbf{x})$) and a sub-region Ω_2 with bad samples (low $f(\mathbf{x})$). Compared to LaMCTS, `PlaLaM` uses a *latent space* and reduces the dimensionality of the search space. **(b)** Sampling follows the learned recursive space partition, focusing on good regions while still exploring bad regions using UCB. `PlaLaM` uses the *maximum* of the sampled value in a region ($\max_{\mathbf{x}_i \in \Omega} f(\mathbf{x}_i)$) as the node value, while LaMCTS uses the mean. **(c)** Upon reaching a leaf, new data points are sampled within the region and the space partition is relearned.

Project Abstract

Path planning, the problem of efficiently discovering high-reward trajectories, often requires optimizing a high-dimensional and multimodal reward function. Popular approaches like CEM and CMA-ES greedily focus on promising regions of the search space and may get trapped in local maxima. DOO and VOOT balance exploration and exploitation, but use space partitioning strategies independent of the reward function to be optimized. Recently, LaMCTS empirically learns to partition the search space in a reward-sensitive manner for black-box optimization. In this paper, we develop a novel formal regret analysis for when and why such an adaptive region partitioning scheme works. We also propose a new path planning method PlaLaM which improves the function value estimation within each sub-region, and uses a latent representation of the search space. Empirically, PlaLaM outperforms existing path planning methods in 2D navigation tasks, especially in the presence of difficult-to-escape local optima, and shows benefits when plugged into the planning components of model-based RL such as PETS. These gains transfer to highly multimodal real-world tasks, where we outperform strong baselines in compiler phase ordering by up to 245% and in molecular design by up to 0.4 on properties on a 0-1 scale. Code is available at https://github.com/yangkevin2/plalam.

Key updates from proposal

We tried the proposed algorithm on various tasks including: synthetic navigation tasks, compiler optimization and modular design with desired properties. PlaLaM achieves the state-of-the-art results on all these tasks.

We have submitted this work to NeurIPS 2021 where the paper is currently under review