Speeding up POMDP Planning via Simplification - Supplementary Material

Ori Sztyglic[1] and Vadim Indelman[2]
[1]Department of Computer Science [2]Department of Aerospace Engineering
Technion - Israel Institute of Technology, Haifa 32000, Israel
ori.sztyglic@gmail.com, vadim.indelman@technion.ac.il

I. Proof for Theorem 1

$$
\begin{aligned}
&-\sum_i w_{k+1}^i \cdot \log \left[\mathbb{P}(z_{k+1} \mid x_{k+1}^i) \sum_j \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j \right] = \\
&-\sum_{i \in A_{k+1}^s} w_{k+1}^i \cdot \log \left[\mathbb{P}(z_{k+1} \mid x_{k+1}^i) \sum_j \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j \right] - \sum_{i \in \neg A_{k+1}^s} w_{k+1}^i \cdot \log \left[\mathbb{P}(z_{k+1} \mid x_{k+1}^i) \sum_j \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j \right] \\
&\geq -\sum_{i \in A_{k+1}^s} w_{k+1}^i \cdot \log \left[\mathbb{P}(z_{k+1} \mid x_{k+1}^i) \sum_j \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j \right] - \sum_{i \in \neg A_{k+1}^s} w_{k+1}^i \cdot \log \left[m \cdot \mathbb{P}(z_{k+1} \mid x_{k+1}^i) \cdot \sum_j w_k^j \right] = \\
&-\sum_{i \in A_{k+1}^s} w_{k+1}^i \cdot \log \left[\mathbb{P}(z_{k+1} \mid x_{k+1}^i) \sum_j \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j \right] - \sum_{i \in \neg A_{k+1}^s} w_{k+1}^i \cdot \log \left[m \cdot \mathbb{P}(z_{k+1} \mid x_{k+1}^i) \right]
\end{aligned} \tag{1}
$$

$$
\begin{aligned}
&-\sum_i w_{k+1}^i \cdot \log \left[\mathbb{P}(z_{k+1} \mid x_{k+1}^i) \sum_j \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j \right] = -\sum_i w_{k+1}^i \cdot \log \left[\sum_{j \in A_k^s} \mathbb{P}(z_{k+1} \mid x_{k+1}^i) \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j \right] - \\
&-\sum_i w_{k+1}^i \cdot \log \left[1 + \frac{\sum_{j \in \neg A_k^s} \mathbb{P}(z_{k+1} \mid x_{k+1}^i) \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j}{\sum_{j \in A_k^s} \mathbb{P}(z_{k+1} \mid x_{k+1}^i) \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j} \right] \leq -\sum_i w_{k+1}^i \cdot \log \left[\sum_{j \in A_k^s} \mathbb{P}(z_{k+1} \mid x_{k+1}^i) \mathbb{P}(x_{k+1}^i \mid x_k^j, a_k) w_k^j \right]
\end{aligned} \tag{2}
$$

II. Pruning tree branches using reward bounds

Usually when planning into the future a planning tree, or a belief tree in the more general case, is built in some manner. This tree approximates the expectation of cumulative future rewards given different possible policies. In order to decide which action should be taken at the root of the tree, rewards should be summed bottom up (leafs to root). This weighted summation for the different routes in the tree, is nothing but the objective function (1). Once the rewards are propagated up the tree, the action (at the root) that present greater future cumulative reward should be chosen, i.e. choose the most promising subtree of the original tree (illustration in Fig. 2a). Due to the recursive nature of (1), (4) this formulation is also recursive and is applied in each belief node of the belief tree. I.e., in each node we propagate up the action that has the biggest corresponding subtree cumulative reward. Thus we get the optimal policy.

A possible way to improve this setting is bounding the tree branches. Meaning, each belief node b_{k+j} in the belief tree has children subtrees corresponding to the different actions that can be taken from b_{k+j}. Each child subtree has it's own upper and lower bound $\{\mathcal{LB}^m, \mathcal{UB}^m\}_{m=1}^{|\mathcal{A}|}$ that we somehow got. So, according to the bounds, when some actions (subtrees) seem to be less promising than their sibling action, we can avoid expanding this tree branch in the first place. Though this approach is sub optimal according to [1]. An alternative way to speedup the process is eliminating existing branches (subtrees or actions) according to these bounds. It becomes possible when for two sibling subtrees m', m'' corresponding to two different actions, we get $\mathcal{LB}_{m'} > \mathcal{UB}_{m''}$ or $\mathcal{LB}_{m''} > \mathcal{UB}_{m'}$. E.g., in Fig. 2c the lower bound of π'''' is higher than all other actions upper bounds. However this becomes problematic if (a) the bounds are not cheaper to calculate than the original objective of some tree. (b) We cannot eliminate all actions but one since the bounds are not tight enough. E.g. in Fig. 2b one cannot say for sure that policy π'''' is better than policy π'' since the latter upper bound is higher than the former lower bound.

This research was supported by the Israel Science Foundation (ISF) and by a donation from the Zuckerman Fund to the Technion Center for Machine Learning and Intelligent Systems (MLIS).

III. Adaptive Simplification Illustrative Example

Consider Fig. 3b and assume the subtrees to b_i^1 were solved using simplification levels that hold $s^2 = s^1 + 1, s^2 < s^3, s^4$. Further assume the immediate reward simplification is $s = s^1$. According to definitions above this means that for b_i^1, $s^{j=1} = \min\{s^1, s^{l=1}, s^{l=2}\}$ and $s^{j=2} = \min\{s^1, s^{l=3}, s^{l=4}\}$. Now, we consider the case the existing bounds of the subtrees were not tight enough to prune, we adapt simplification level of the tree starting from $b_i^1 : s^1 \rightarrow s^1 + 1$. Since $s^1 < s^1 + 1$ we re-simplify the subtree corresponding to simplification level of s^1 to simplification level $s^1 + 1$, i.e. to a finer simplification.

However we do not need to re-simplify subtrees corresponding to s^2, s^3, s^4: The tree corresponding to s^2 is already simplified to the currently desired level thus we can use its existing bounds. For the two other trees, their current simplification levels, s^3 and s^4, are higher (finer) than the desired $s^1 + 1$ level, and since the bounds are tighter as simplification level increases we can use their existing tighter bounds without the need to 'go-back' to a coarser level of simplification. If we can now prune one of the actions, we keep pruning up the tree. If pruning is still not possible, we need to adapt simplification again with simplification level $s^1 + 2$.

IV. ALGORITHMS

Algorithm 1 Prune Branches

procedure PRUNE
  Input: (belief-tree root, b; bounds of root's children, $\{\mathcal{LB}^m, \mathcal{UB}^m\}_{m=1}^{C}$) ▷ C is the number of child branches going out of b.
  $\mathcal{LB}^\star \leftarrow \max_m \{\mathcal{LB}^m\}_{m=1}^{C}$
  for all children of b **do**
    if $\mathcal{LB}^\star > \mathcal{UB}^m$ **then**
      prune child m from the belief tree
    end if
  end for
end procedure

Algorithm 2 Simplified Information Theoretic Belief Space Planning (SITH-BSP)

procedure FIND OPTIMAL POLICY(belief-tree: $\mathbb{T}$)
  $s \leftarrow s_0$
  return ADAPT SIMPLIFICATION($\mathbb{T}$,s)
end procedure
procedure ADAPT SIMPLIFICATION(belief-tree: $\mathbb{T}$, s_i)
  if $\mathbb{T}$ is a leaf **then**
    return $\{\mathbf{lb}, \mathbf{ub}\}$ ▷ Corresponds to immediate reward bounds over the leaf (5).
  end if
  Set simplification level: $s \leftarrow s_i$
  for all subtrees $\mathbb{T}'$ in $\mathbb{T}$ **do**
    ADAPT SIMPLIFICATION($\mathbb{T}'$,s)
    Calculate $\mathcal{LB}^{s^j}$,$\mathcal{UB}^{s^j}$ according to s and (12)
  end for
  Using $\{\mathcal{LB}^{s^j},\mathcal{UB}^{s^j}\}_{j=1}^{|\mathcal{A}|}$ and Alg. 1 prune branches
  while not all $\mathbb{T}'$ but 1 in $\mathbb{T}$ pruned **do**
    Increase simplification level: $s \leftarrow s + 1$
    ADAPT SIMPLIFICATION($\mathbb{T}$,s)
  end while
  Update $\{\mathcal{LB}^{s^j\star},\mathcal{UB}^{s^j\star}\}$ according to (15)
  return optimal action branch that left $a^\star$ and $\{\mathcal{LB}^{s^j\star},\mathcal{UB}^{s^j\star}\}$.
end procedure

V. Additional Entropy Results

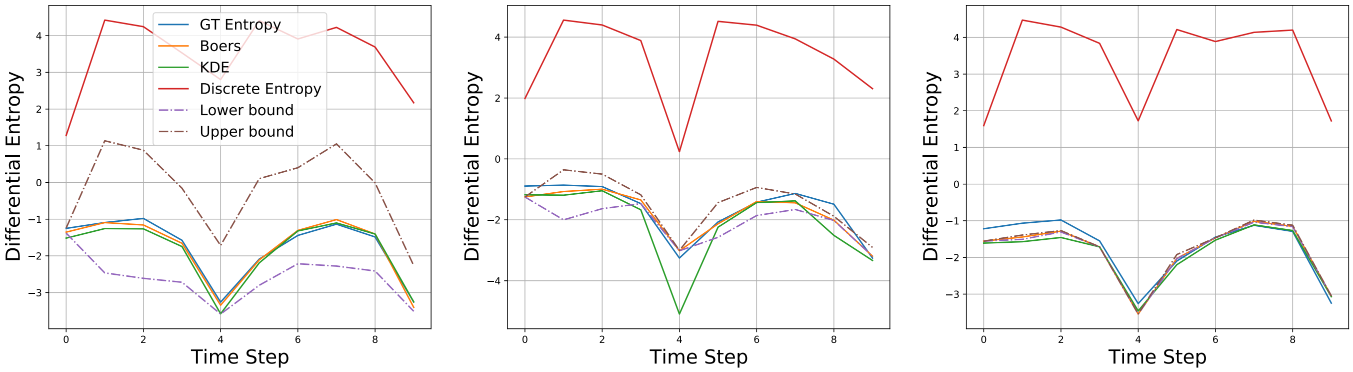

Fig. 1: Differential Entropy Approximations ans Bounds. Calculations were done using 100 particles. From left to right: Simplification is $N^s = \{0.1, 0.5, 0.9\} \cdot N$

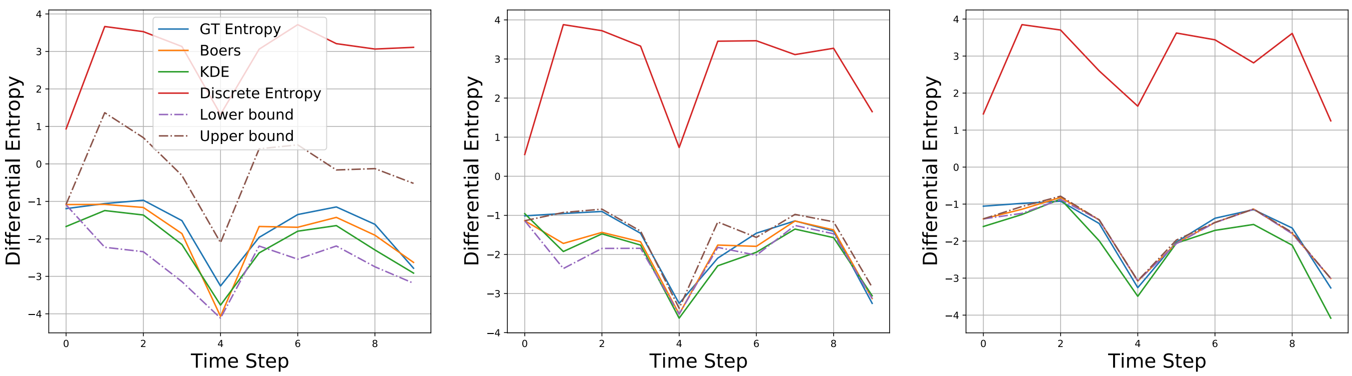

Fig. 2: Differential Entropy Approximations ans Bounds. Calculations were done using 50 particles. From left to right: Simplification is $N^s = \{0.1, 0.5, 0.9\} \cdot N$

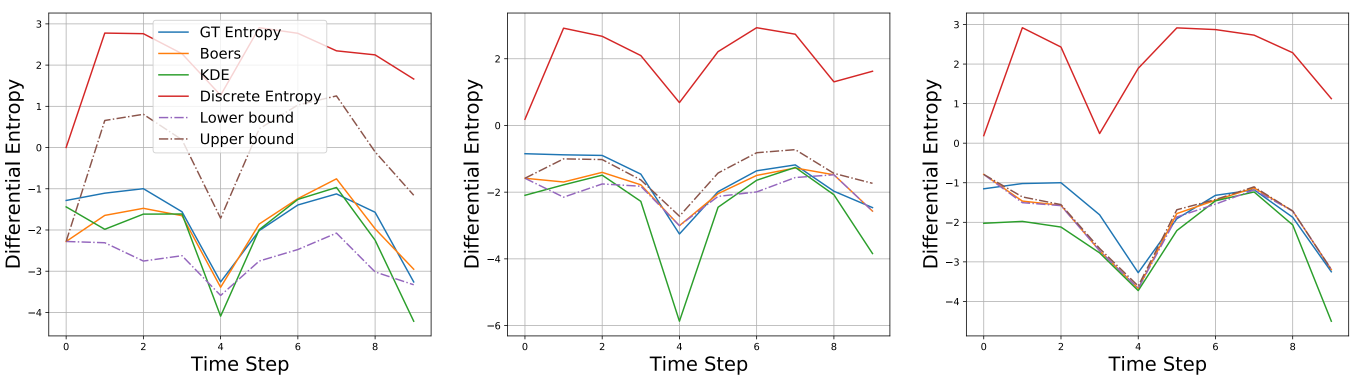

Fig. 3: Differential Entropy Approximations ans Bounds. Calculations were done using 20 particles. From left to right: Simplification is $N^s = \{0.1, 0.5, 0.9\} \cdot N$

References

[1] Michael H. Lim, Claire Tomlin, and Zachary N. Sunberg. Sparse tree search optimality guarantees in pomdps with continuous observation spaces. In *Intl. Joint Conf. on AI (IJCAI)*, pages 4135–4142, 7 2020.