Response Letter

May 2, 2019

We thank the reviewers and editors for their helpful comments to improve the paper.

Following, we explain how the four remarks pointed in the editors' comments have been addressed.

> That your learning problem can be solved by planning does not mean it it PSPACE-complete; only that it is a member of PSPACE. To show PSPACE-hardness, you would have to show that arbitrary planning problems can arise in your setup. You either need to provide an argument for the latter, or to reformulate our text removing the claim of PSPACE-completeness and merely pointing out that SAT approaches are no longer obvously applicable given the lack of a length bound.

We agree with the editors' observations as well as reviewer 1's remarks about the complexity of the learning task. Certainly, the fact that the learning task is solved via planning does not make it a PSPACE-complete task; in order to prove this, we should indeed show a planning problem is reducible to a learning problem. Actually, our intuition is that both tasks, learning and planning, are equivalent, and that reducibility is applicable in both directions. Nevertheless, this is an issue we must still study in depth and we expect to tackle it in the near future.

For the purpose of this paper, we have removed any reference to PSPACE-completeness. We simply highlight that when the length of the plan trace is unknown, we can no longer leverage the advantages of using a SAT solver as the large majority of the current learning approaches do. This is actually the key contribution of this paper: the minimal observation needed by FAMA to learn an action model is a plan trace composed solely of a full observable initial state and a partially observable final state (goal state).

Clarifications regarding this remark are highlighted in red in page 11. We also rephrased some sentences of the section Conclusions.

> It needs to be made clearer whether and how ordering constraints between actions and states are handled in your compilation scheme.

This comment along with the next one are two insightful observations about the compilation scheme. Actually, we do have a mechanism in FAMA to tackle both issues, although we did not explain it in the previous version of the document because we considered it was a rather technical issue. We have now fixed this.

The set F_Λ includes a fluent $action_{applied}$ to force the execution of at least one action between two observed states (page 13). Then, when validating a plan trace that contains an interleaved sequence of actions and states, the `validate` actions include an extra precondition, at_{i+1}, where i is the index of the last observed action before the state we are validating. This additional precondition ensures that applied actions are also ordered with respect to the observed states. This is highlighted in red in page 15.

It needs to be made clearer whether and how action/state insertion can be forbidden in your compilation scheme when part of every state is observed.

When at least a fluent of every state is observed in the plan trace, we need to ensure that only one action is executed between two observed states. This is done by adding the precondition $\neg action_{applied}$ in every `apply` action.

In the case of a fully known plan trace (every action is observed and every state too even partially), the conditional effects formed by fluents F_π and at_i are transformed into a regular precondition+effect in the `apply` actions. This mechanism prevents the planner from executing `apply` actions other than the ones that appear in the plan trace.

This is highlighted in red in page 14.

The term GTM must be explained and properly discussed.

We agree with the remarks point out by reviewer 3. We borrow the definition of GTM provided by the reviewer (highlighted in red in page 5). Likewise, in section 5, we clarify that approaches that use the GTM as the reference point to compare the learned models, use plan traces generated with the GTM; i.e., they use the domains of the IPC benchmarks.