Iterative Planning for Deterministic QDec-POMDPs, Response to ICAPS reviews

First, we would like to thank the reviewers for their detailed comments. We feel that you somewhat underestimate our contribution. (Q)Dec-POMDPs are very hard to solve, and we show many orders of magnitude scaling up over previous solved problems. Specific answers below.

Reviewer 1:

1. Comparison with Dec-POMDP solvers – the solvers that we compare against are not optimal, although they try to find better policies (in terms of E[C]). This is why their results vary, not because they are anytime and were stopped early. They were run to convergence.
2. Time propositions – we only add propositions of type (time ?t), in order to force specific actions to occur at time t. We do not add copies of all propositions for each time step. Grounded actions do have one copy per time step.
3. Analyzing task allocation – this is a good suggestion that we will improve for the final version. Results in problems with many re-planning episodes (last column) are due to task allocation improvements. We will add a chart showing the decrease in E[C] as a function of the number of replanning.
4. Ensuring that replanning will not repeat the previous solution is done by adding a constraint forcing the agent that created the failed constraint to achieve the required proposition later than it did originally. This is not thoroughly explained, but appears in the last paragraph before 3.5.
   * Yes, but only for collaborative actions
5. Horizon – it is unclear why some reviewers thought that we are using a finite horizon – we do not (see paragraph above definition 2). For the Dec-POMDP planners that require an horizon we tried all horizons and took the one where the Dec POMDP planner performed the best.
6. Table 4 – explained in the last paragraph of section 4. We will improve this.
7. Allowing actions after t\_d – t\_d represents the time of the last timed constraint. This does not mean that planning must stop afterwards. T\_inf represents that we do not care about the exact time after the last constraint.
   * I think we should remove t\_inf at all. It makes it harder to understand why we need it, and it has not contribution to the work.
8. Actions with non-unit durations are outside the scope of this paper (nor are they defined currently in the QDecPOMDP paper). Significant adjustments are probably needed to support such domains.
9. With thank you for the many minor comments that will be fixed.

Reviewer 2:

1. Not clear what the algorithm is trying to compute – as explained in the original QDec-POMDP paper (section “Belief States and Joint Policies”) one can either compute a joint policy tree, or local trees for each agent. The original QDec paper computed a joint policy tree, and we choose the latter, computing an individual tree for each agent. This is explained in section 2.1. If your comment refers to what constitutes a valid solution – we look for a satisfying solution, where every leaf is a goal state, as is done in contingent and classical planning.
2. Soundness and completeness – POMDPs and Dec-POMDPs are very difficult to solve, and research in these areas often suggests algorithms with no formal optimality guarantees, but manage to solve larger problems. We take a similar approach, suggesting an algorithm that has no guarantees, but scales up well in experiments, many orders of magnitude beyond current Dec-POMDP solvers, as we show. We do explain how the policy soundness can be validated before execution. Your suggestion for first presenting a sound algorithm and then relaxations may be useful.
3. Mathematical presentation – in this conference paper with very limited space we choose to present the algorithm in a less precise manner, in the hope of allowing more space for examples and explanations. Some people prefer a more formal presentation, but this is the way that many ICAPS papers are presented. The presentation style that you suggest, with clear mathematical definitions, a clean sound (and complete?) algorithm, and algorithmic relaxations, without reducing examples and explanations will have to wait for the journal version (following acceptance).

Reviewer 3:

1. Definition of QDec-POMDP – we are not the first to suggest this model (see the AAAI 2013 paper). You argue that as there are no probabilities, the name should not contain MDP, and you may be right. The AAAI 2013 shows, though, that QDec-POMDP share the same complexity class as Dec-POMDPs, as well as the same solution style (policy trees). Regardless of the right term for this problem type, decentralized multi agent planning under partial observability is clearly and interesting (and difficult) problem.
   * We must examine what are the differences between our version of QDec-POMDP (without non-determinisem, and probabilities) and other planners.
2. Communication – communication is certainly possible, but the model must contain explicit communication actions. That is, communication must be made explicit, as part of the policy. We assume no implicit communication, as is typical in Dec-POMDPs. Our example domains indeed contain no explicit communication actions, and it is interesting to add such domains to the benchmark set.
3. Action contingent on feature f – the example is unclear to us. The agents agree before planning on a policy, which may contain such communication (e.g., agent 1 drops a coin, and if agent 2 sees the coin, it goes left rather than right). However, another, third agent may interrupt the communication by picking the coin. This is an example of the unsoundness of our approach.
   * 2 + 3: I think he talks about what we discussed on the phone, when agent 1 observes something and then another agent changes the world without agent 1 knowing about it. – (falsification)
4. Algorithmic details – we choose to sacrifice precise mathematical formulation and complete pseudo-code in favor of explanations and examples (see also response 3 to reviewer 2).
5. Simulating plan trees – this is of course exponential, but is not a problem with the problem sizes that we currently solve. In Dec-POMDPs scaling up is very difficult, and it is not clear that simulating the policy is the bottleneck, rather than finding a policy to begin with.
6. E[C] indeed required a probability, but we had to define probabilities for the Dec-POMDP models (we used a uniform probability over the initial state). For IMAP, we did as you said. This is why no E[C] is computed for Tables 3+4.
7. Thank you for your minor comments.

Reviewer 4:

1. Clearly, reachability analysis is difficult. We instead use a collaborated delete relaxation forward expansion. We will allocate more space to explain this in the final version.
2. Constraint removal – when a constraint c inserted by agent i cannot be achieved, we backtrack to i, removing all constraints introduced by all agent after c was inserted by i.
3. falsifying preconditions – we are working on a method (not described in this paper, mainly due to the lack of space) for adding constraints forcing other agents to maintain required preconditions of other agents. Other agents do not need to achieve these literals, just avoid destroying them prior to a given time. This never happens on current benchmarks.