Reviewer 1:

|  |  |
| --- | --- |
| *Significance:* | **2**: (modest contribution or average impact) |
| *Soundness:* | **3**: (correct) |
| *Scholarship:* | **2**: (relevant literature cited but could be expanded) |
| *Clarity:* | **3**: (well written) |
| *Reproducibility:* | **3**: (authors describe the implementation and domains in sufficient detail) |
| *Overall evaluation:* | **1**: (weak accept) |
| *Reviewer's confidence:* | **2**: (medium) |
| *Suitable for a demo?:* | **1**: (no) |
| *Nominate for Best Paper Award:* | **1**: (no) |
| *Nominate for Best Student Paper Award (if eligible):* | **1**: (no) |

The paper introduces a new algorithm to solve multi-agent problems whose underlying model is a Qualitative Dec-POMDP, where probability distributions are substituted by sets of states. It is assumed that action effects and observations are deterministic, a fixed horizon is given, actions occur in a unit timeline (and cost, although it is claimed that it can be easily relaxed, but not sure how scalability would suffer) and that the initial belief state is shared among all agents. The proposed algorithm iterates over each Agent, generating a policy tree using a contingent planner. It then moves on into the next agent, incorporating constraints to make sure that the policy tree of the next agent makes a valid joint policy tree that is executable for all agents, leading to a goal leaf state in the policy of each agent. The algorithm incrementally builds the policy tree for all agents. If one agent cannot accommodate the constraints enforced by other agents, then the problematic constraint is detected and the algorithm back-jumps to the agent that produced it, and enforces that the constraint is relaxed by increasing the slack. Constraints are made to synchronize joint actions, as well as making sure that preconditions that are unreachable for an agent are reached by other agents before they are needed. The paper then reports experiments in two domains, a 1 dimensional grid with heavy boxes that have to be pushed up among 2 agents, and small boxes that do not require cooperation. Uncertainty comes at the initial location of the boxes, either at the goal (out of the grid), or at one of the locations. The other domain is a rovers domain with rocks sampling requiring 2 agents. Uncertainty comes in the waypoints measurement equipment needed. The planner is compared with other Dec-POMDP solvers and scales up better, although the other planners try to optimize solutions instead of just finding a satisficing policy, and hence might not be a good comparison as mentioned in the paper.

* Comparison with optimal algorithms: As written in section 4, DICE algorithm which we compared with is not an optimal planner, JESP is also a method for looking for a local optimum.

The algorithm it's a practical contribution to help improving the scalability of QDec-POMDP solvers, and opens up a new direction on an alternative search over the state space, exploiting the inherent structure in the factored representation of the problem to leverage contingent planners performance.

Below I list comments, some minor, some highlighting concerns on different parts of the paper:

After definition 1, the assumptions of deterministic actions are laid out, so it would be interesting to make explicit that this assumptions are key for mapping each agent problem from a QDec-POMDP into a contingent problem.

Right before section 2.1, it's nfot clear why heterogenous initial belief states is a feature that appears only in on-line planning. Further explanation is required to convey the argument.

Page 3, col 1, vecq -> missing math format

* ‘\’ was missing, fixed it

Example 1: missing Heavy\_j in def of P. In the same example the use of "truly" is confusing.

* If not clear, the predicate Heavy\_j describes that the box j is heavy and thus requires two agents to be at location j to push the box together. (~Heavy = light)

In Fig 1, it's not clear what the numbers 1 to 18 mean. No-op is not understood until section 3.2, which makes the tree confusing for the first part of the paper.

* These are the indexes of the nodes so we could relate to them in text. We need to add this to the description of the figure, just for clarification. Also, add to the figure description what no-op means.

End of 2nd paragraph in section 3.2, it's important to clarify that agent 2 has to observe at the same time t\_0 as agent 1.

* He doesn’t!

First paragraph, page 5, Figure 2 -> Figure 3

* Fixed

Section 3.3, Would be great to clarify how time t affects the compilation. Not clear if every action and literal is copied for each t between t\_0 and t\_d.

* No it doesn’t – only one proposition per time

Section 3.4, Why don't you fail directly once constraint c is identified? How do you ensure that the solution will be different the second time you try to solve the problem? Can see that recognizing c it's important for doing back-jumping, but not sure about the extra try on solving. Specially as this insight cannot be extracted from the experiments, and one would imagine that the planner would take a long time to fail, unless the contingent planner is not complete, which should also be stated as a source of incompleteness.

* Last paragraph before 3.5

The experiments don't back-up the claim that the algorithm does task allocation to reduce cost over completion time, so the claim in the introduction is not supported beyond proposing a method for doing so, but not testing if it's actually practical. Any of the reported instances do sub-goal allocation optimization? What's the total improvement in terms of E[C]? This data can help justifying the claim.

* The meaning of the claim in the introduction was that the overall time is reduced. Optimal solvers such as GMAA-ICEs, will surly provide much better E[C] but will fail at bigger problems. Managing to provide results is better than none – task allocation allows IMAP to reduce E[C] even more.

In the experiments, It should be stated how the expected cost E[C] is extracted, and also mentioned in the table caption. Tables are reported but no insights are given in terms of IMAP algorithm design. For example, I'd like to know if the last 2 instances have more planning calls because there was a back-jumping, or because there was some sub-goal optimization, and if so what's the gain in terms of E[C].

* That is an interesting thought – about how to display the backtracking information – perhaps using visualization.

Furthermore, It should be clarified what's reported for the other anytime planners, it's not clear if it is first solution, or the last solution they compute, assuming that they are not optimal, as they stop early and disagree about optimality. Would be beneficial to clarify these facts about these planners.

* The other planner were used as a Blackbox, and to look into each planner’s output, was not an option since other planners do not provide other than the expected cost.

The horizon used in the experiments is omitted, If it doesn't affect the scalability of IMAP, this should be mentioned, but at least it has an impact on the size of the compilation, so some discussion is expected.

* Horizon didn’t affect IMAP’s result as far as we know for the small problems that we compared with. Checking the scalability of the planner with higher horizon is another aspect to be checked, and left for further examining in the future.

Table 4, columns 4-6 numbers are hard to understand.

* Agreed, description should have been added. Though it is explained in the last paragraph of section 4.

Question:

Why do you need to allow actions after horizon d, between t\_d and t\_inf? What does it mean for an action to happen between t\_inf and t\_inf? The semantics are not clear. This is mentioned in section 5.

* Between t\_d and t\_inf collaborative actions cannot be performed. Thus one can also consider that defining t\_d = t\_inf in-order to define more rigid model definition.

How are you going to get the compilation of non-unit time? For that case Metric-FF wouldn't be able to solve the problem. Is it a matter of just using a Temporal Planner?

* ?

Which horizons were used for the experiments? once constraint c is identified, How many times the extra computation helped? Does the contingent planner fail quickly when no plan is found?

* In our case – horizon used was 100.
* I’m not certain which contingent planner is talked about – if ours – there are some timeouts from the Metric-FF that can be merge – this is a problem we have to deal with.
* Each time the planner fails – IMAP changes the model and thus in some sense – learns something new – so it helps.

Reviewer 2:

|  |  |
| --- | --- |
|  | Review 2 |
| *Significance:* | **2**: (modest contribution or average impact) |
| *Soundness:* | **1**: (major errors) |
| *Scholarship:* | **3**: (excellent coverage of related work) |
| *Clarity:* | **1**: (hard to follow) |
| *Reproducibility:* | **2**: (some details missing but still appears to be replicable with some effort) |
| *Overall evaluation:* | **-1**: (weak reject) |
| *Reviewer's confidence:* | **2**: (medium) |
| *Suitable for a demo?:* | **1**: (no) |
| *Nominate for Best Paper Award:* | **1**: (no) |
| *Nominate for Best Student Paper Award (if eligible):* | **1**: (no) |

Dec-POMDPs are known models for POMDP planning involving multiple agent, where the joint policy or solution is computed globally, not by each agent. Q-Dec-POMDPs, are a variation of Dec-POMDPs where uncertainty is represented by sets of states, and not probability distributions, and where a goal is to be achieved jointly. In a AAAI-2013 papers, it was show how to exploit translations and single-agent contingent planners to solve Q-Dec-POMDPs.

In this work, the authors deal apparently with the same model, but look for a different type of solutions, where each agent computes its own control in a distributed way. This is clearly more challenging and interesting. The authors propose an algorithm for computing such control in a distributed way, called IMAP, and evaluate it over some benchmarks.

The main problem that I find with the algorithm and the paper is that it's not clear what the algorithm is trying to compute,

* Individual local trees

and algorithm that looks is not even sound (although solutions can be checked for soundness). I think that two key missing components in the paper are:

A -- a clear mathematical definition of what IMAP is aimed at solving B -- a proof that the algorithm is sound wrt that definition, or a clear explanation of why it is not, and why it cannot be rendered sound (except by checking the solution explicitly, which doesn't speak well about the algorithm in the first place).

The first should follow Defs 1 and 2. If the authors need to introduce the definition and then a basic algorithm that is sound, and hence addresses the problem, but which is not sufficiently effective, fine. Then, the authors could introduce suitable approximations and shortcuts that sacrifice soundness for efficiency, while checking the soundness of the solutions explicitly. This will make the paper cleaner and much easier to follow and build upon.

Possibly readers are expected to fill out the details of A by themselves from the text in the paper, but this is no good (at least for this reader). In any case, don't give up on me, and try to address this in the rebuttal.

* The main reason that IMAP is not mathematically defined is because it is an algorithm that is meant to solve modified QDec-POMDP problems. We do however, state clearly that the modified models are not guaranteed to have a solution.

Reviewer 3:

|  |  |
| --- | --- |
|  | Review 3 |
| *Significance:* | **1**: (minimal contribution or weak impact) |
| *Soundness:* | **2**: (minor inconsistencies or small fixable errors) |
| *Scholarship:* | **3**: (excellent coverage of related work) |
| *Clarity:* | **2**: (mostly readable with some room for improvement) |
| *Reproducibility:* | **2**: (some details missing but still appears to be replicable with some effort) |
| *Overall evaluation:* | **-2**: (reject) |
| *Reviewer's confidence:* | **3**: (high) |
| *Suitable for a demo?:* | **2**: (maybe) |
| *Nominate for Best Paper Award:* | **1**: (no) |
| *Nominate for Best Student Paper Award (if eligible):* | **1**: (no) |

This paper describes an iterative planning algorithm for qualitative decentralized partially observable markov decision processes.

More specifically, I believe that this covers deterministic multi-agent planning problems with partial observability. I'm not entirely sure what it means to have a qualitative POMDP, since there is no probability measure, only simple non-determinism, and no utility function. Also, in this case the paper limits itself to actions with deterministic outcomes, and finite horizons. I'm not sure how helpful it is to refer to something without the key features of probability and utility as an MDP.

* Not sure about the rest, but surly it is not finite horizon. Guy?

Another limitation is that communication seems not to be possible -- on p. 4 we are told that agent 2 must determine the state of agent 1 by using its own observations and inference about what actions agent 1 will pursue, based on facts it has observed (i.e., those facts that agent 1 will use to direct choices in its policy). It wasn't clear to me how this would work if, say, agent 1 could use feature f to decide what action to take in a state, but some agent could change the value of f between when agent 1 chose and agent 2 observed.

* ???

I found the details of the algorithm quite difficult to follow, since much of it is discussed in conversational terms in the body of the paper, and only summarized formally in the pseudocode of Algorithm 1. I would have preferred to get pseudo-code or mathematical formalization for more of the details. If I were to try to implement this myself, I would have to read carefully over the text and translate it into such pseudo-code myself.

This left me not confident that I could replicate the method, nor that I understood how the different pieces of the algorithm -- which uses a number of more or less discrete mechanisms to perform its overall function.

* We could put more technical information into the paper.. ( for replication )

In "Ensuring Soundness" we are warned that the procedure "may not produce a sound, executable plan." But that "It is simple... to ensure that the resulting joint plan tree is valid, by simulating the plan tree, and checking that in each feasible branch, all preconditions are met when actions are executed, and that in each leaf the goal is achieved. We take this approach in ensuring that the plan trees generated in our experiments are indeed sound." This may be \*conceptually\* simple, but it certainly is not a practical procedure for any but the tiniest problems.

* I would love to hear what are the tiniest problems the reviewer deals with

I would like a much stronger argument to convince me that I should trust this method to find me a good policy for a problem I had. It seems like the user would have to take it on faith that he or she is likely to get an acceptable answer, with no guarantee. For now this uncertainty keeps me from being able to give the paper a high significance score. I am not necessarily saying that the work is insignificant, just that the paper does not clearly and persuasively make a case for its significance.

How would these problems change if you

* ?

Some minor issues of presentation:

\* Please put the definition of QDec-POMDP into the abstract. When this paper is collected for reference, readers who only see the abstract should get the meaning of the abbreviation.

* Noted

\* The effects function is not typeset correctly. Latex interprets your eff(a\_i) as "e \* f \* f(a\_i)" and typesets it accordingly. Try \operatorname{eff} or something similar to get LaTeX to do the right thing.

* Guy?

\* The tables are not adequately explained. In table 2:

\* What is E[C]? Ordinarily, I would have assumed that this was expected cost, but there's no indication how expectation would be computed in a domain without a probability measure (or, for that matter, a cost function). Did you run IMAP on a DecPOMDP, treating it as qualitative, and then compute an expected cost based on the probabilities (which were used by the other algorithms)?

* E[C] in our case is the average depth of the joint tree.
* Not sure I understand how is it possible to run IMAP on DecPOMDP?

\* I'm pretty sure that "Planning" is the number of single agent planning episodes, but you should say so explicitly.

* True

\* Also, if we are meant compare the different alternatives, Compilation, GMAA-ICE, etc., please help us out by indicating which of the approaches offers the best run-time and E[C] for each problem. A simple method would be to shade or bold-face the relevant cell in the table.

Reviewer 4:

|  |  |
| --- | --- |
|  | Review 4 |
| *Significance:* | **2**: (modest contribution or average impact) |
| *Soundness:* | **1**: (major errors) |
| *Scholarship:* | **3**: (excellent coverage of related work) |
| *Clarity:* | **3**: (well written) |
| *Reproducibility:* | **3**: (authors describe the implementation and domains in sufficient detail) |
| *Overall evaluation:* | **-1**: (weak reject) |
| *Reviewer's confidence:* | **4**: (expert) |
| *Suitable for a demo?:* | **1**: (no) |
| *Nominate for Best Paper Award:* | **1**: (no) |
| *Nominate for Best Student Paper Award (if eligible):* | **1**: (no) |

The paper is fairly clear, with one exception (my first question below) There also seem to be some serious problems with the method which have not been solved.

The authors state:  
"In addition, we identify a set P- of non-constant propositions that none of the independent or collaborative actions of agent i can achieve, yet appear in a precondition of an action a in A+ or in the goal G."  
How is such a set identified? Determining whether or not an arbitrary fact can be satisfied is itself a planning problem. It reads as though goals that cannot be achieved are removed prior to planning -- how is it known that they cannot be achieved?

* If I understand the question correctly, these are the artificial actions: an agent 1 in his plan, achieves a specific by doing some actions. He gets a precondition and as a result this creates a new action for agent 2 that we call “artificial action X” that has the negated preconditions but same effects. (agent 2 must act with the original action or the artificial one, depending on what branch he is). This set is usually one precondition which was achieved by observing something and acting upon it. It can be taken from the “observed” field of the action.

What strategy is used to remove constraints from the problem when backtracking? Can you comment on what kind of inference might be possible to remove the "correct" constraints that can be better achieved in an earlier iteration?

* It is good idea to start with the earlier conflicts first. Once these are agreed upon both agents, hey can continue with the planning procedure with later collab actions. If agent 2 couldn’t help with action at time t, agent 1 will try again in time t+1 and will let agent 2 start again. This can be improved by letting agent 2 send a message to agent 1 with the minimal time that he can help with the current conflict.

The issue ofan agent falsifying the required preconditions of another agent seems serious. It sounds as though the approach is only half designed. While the deeper discussion may not fit into the current paper, could you comment on the seriousness of this issue?

* Guy?

Under what kind of domains is this issue most prevalent (is it simply a case of domains in which there are more delete effects?) and in the case when the combined policy is not sound, how is a sound policy derived? Can new constraints be added to the initial problem which will correct the plan?

* The idea of adding “no-op” actions into a unsound joint plan tree is already the attempt to correct the joint-plan.

An unsound method can still be useful, but I would prefer that there is some discussion on its applicability.