Author's Reply.

Ryohei Oura, Ami Sakakibara, and Toshimitsu Ushio

January 24, 2020

We are thankful to the reviewers for their fruitful comments. We have revised our manuscript according to the comments. We hope that our revisions have improved the manuscript to their satisfaction.

In the following, we refer to a transition-besed limit-diterministic generalized Büchi automaton as tLDGBA and a non-generalized one as tLDBA in our replies.

1 Reply to Reviewer 1 (19781)

Review Point 1.1. My primary issue with this paper is in the claim of the paper's main result, Theorem 1. In itself, the claim of Theorem 1 has nothing to do with automata or learning. It states that we consider an MDP M and any LTL formula phi, and appears to say the following: if there exists a stationary (or, in the author's words, positional) policy satisfying phi, then a policy maximizing the expected discounted reward (for some discounted factor) will actually be a policy that satisfies phi.

If my reading of the theorem claim is correct, such a result is clearly incorrect. Consider an MDP M with states s_0, s_1 and actions a,b such that $P(s_i, a, s_i) = 1, P(s_i, b, s_{1-i}) = 1$ (in other words, action a results in the agent state not changing, and b results in it changing to the other state). Consider the rewards given by $R(s_0, *, *) = 0, R(s_1, *, *) = 1$; in other words, the agent does not receive anything when it is at s_0 and receives 1 when it is at s_1. Consider now phi = "always s_0".

Clearly, if $s_0 = s_{init}$, there exists a positional policy satisfying phi: it just applies action a over and over again. (Even if s_init is not set, we can consider phi = "next always s_0", and a policy that applies b at s_1 and a at s_0). On the other hand, regardless of the discount factor, the reward obtained by an agent that satisfies phi is by definition 0. The maximal expected reward will actually be achieved by an agent that goes to s_1 and stays there indefinitely, thus seemingly contradicting the theorem claim.

Because this "counterexample" is so obvious, I am tempted to believe that it stems from a misunderstanding of the theorem claim: perhaps the unusually written part saying "any algorithm that maximizes the expected reward ... will find a positional policy" means something else than what I meant? Nonetheless, before continuing with evaluating the paper, I believe that this issue needs to be cleared up.

Reply. For an MDP M, an augmented tLDBA $\bar{B}_\varphi$ corresponding to an LTL formula φ, the product MDP $M^\otimes$ of M and $\bar{B}_\varphi$, and a reward function based on the acceptance condition of $M^\otimes$, we see an optimal policy as an positional policy on $M^\otimes$ maximizing the expected discounted reward. Therefore, in your counterexample, the optimal policy and the reward function should be considered as a positional policy on the product MDP $M^\otimes$ and be designed based on the acceptance condition of $M^\otimes$, respectively. Probably, this misunderstanding of Theorem 1 is due to the lack of expression of our theorem claim. We revised the theorem claim to clarify that the reward function is based on the acceptance condition of $M^\otimes$ and an optimal policy is considered on $M^\otimes$.

Review Point 1.2. The connection between RL-based synthesis and the theoretical results of Section III should be made much clearer.

Reply. Theorem 1 implies that for the product MDP $M^\otimes$ of an MDP M and an augmented tLDBA corresponding to a given LTL formula φ, we can obtain a feasible positional policy satisfying φ on $M^\otimes$ by an algorithm maximizing the expected discounted reward with large enough discount factor if there exists a positional policy on $M^\otimes$ satisfying φ with non-zero probability. We added the explanation of the connection of Theorem 1 and the RL-based synthesis after the proof of Theorem 1.

Review Point 1.3. Apart from the potential theoretical interest, it's not clear why using LDBA would be better for policy synthesis than using other automata; the example only compares the authors' results with another LDBA approach.

Reply. The reason we use LDBAs instead of other automata is mainly described in [1]. It is known that deterministic Rabin automata (DRA) and non-deterministic Büchi automata (NBA) can recognize all of the ω-regular language. However, there is a counterexample of an MDP M and an LTL formula φ with Rabin index 2, which is mentioned in [1], such that, although there is a positional policy satisfying φ with probability 1 on $M^\otimes$ of M and the DRA, optimal policy obtained from any reward based on the acceptance condition of the DRA do not satisfy the LTL formula with probability 1. This is because the reward function is defined for each acceptance pair of the acceptance condition of the DRA, namely the counterexample is due to that only one acceptance pair of the DRA is considered in one learning. Further, LDBAs are not only as expressive as NBAs but also the number of non-deterministic transitions are much less than NBAs. However, in an LDBA, the order of visiting accepting sets of the corresponding LDGBA is fixed. Therefore, the reward function based on the acceptance condition of the LDBA tends to be sparse. We added a remark (Remark 1) to our manuscript that explains the sparsity of rewards for Büchi acceptance and that the sparsity is critical against an RL-based synthesis.

Review Point 1.4. The notation ”section”, which is combined with Section II.A, is unclear and imprecise: what is omega in the exponent, what is a ”scalar bounded reward”, what is s_{init} (it is not mentioned in M), etc.

Reply. The omega in the exponent means the infinite connection, namely Σ^ω means $\Sigma\Sigma\dots$ and $S(\Sigma S)^\omega$ means $S\Sigma S\Sigma S\dots$. s_{init} is the initial state. We revised “immediate scholar bounded reward” to “immediate reward”.

Review Point 1.5. The notion of a formula being ”satisfied” if it is satisfied with any non-zero probability (instead of 1) is counterintuitive.

Reply. The reason we employ the notion that a formula is satisfied with a non-zero probability is to more generally evaluate an obtained policy. Underlying the notion, the goal is to obtain a policy efficiently that maximizes the satisfaction probability.

Review Point 1.6. The introductory section is not clear about the ultimate purpose and contribution of the paper: is it to improve RL performance for LTL specifications? If so, OK, but that should be stated clearly.

Reply. Yes, it is. We revised the introduction in our manuscript to clarify our contribution.

Review Point 1.7. The sentence ”In general, there are uncertainties in a controlled system...” needs rephrasing: maybe something like ”Because of inherent stochasticity of many controlled systems, ...”.

”we model a controlled system” – it is not clear whether this is the authors' contribution or prior work. It might be better to say ”Previously, ...”

Reply to Typo : ”syntehsis”

Reply. We revised it.

2 Reply to Reviewer 5 (19849)

Review Point 2.1. The contribution of the paper is unclear. The authors claim that the proposed algorithm improves the learning performance compared to relevant approaches [1]-[4]; however this is a vague statement. Does this mean that the proposed algorithm is more sample efficient?

Reply. Yes, it is. By the definition of augmentation in our manuscript, the sparsity of rewards is relaxed compared to the case of using tLDBAs. Therefore, our proposed algorithm is more sample efficient compared to the case of using tLDBAs. In addition, the reward function based on the acceptance condition of an augmented tLDGBA does not have to have memory of previous visits to the accepting sets of the original tLDGBA because the augmented tLDGBA keeps track of the previous visits. The reward function defined as the accepting frontier function [2], however, has no memory of previous visits to the accepting sets of the original tLDGBA despite they construct the product MDP from the original tLDGBA. Therefore, there is an example of an MDP M and an LTL formula φ that there exists no positional policy satisfying φ on the product MDP of M and a tLDGBA corresponding to φ.

Review Point 2.2. The last paragraph in the section with the simulations is unclear and possibly wrong. The authors argue that [2] cannot find a policy for the considered example. However, [2] (and [3], [4]) has been shown that if there exists a policy that satisfies the LTL spec, then it will find it. This reviewer's understanding is that [2] is not as sample efficient as the proposed algorithm. In other words, [2] can also find a policy but it requires more episodes. The authors need to clarify this point.

Reply. In [2], they claim that if there exists **a positional policy satisfying an given LTL formula φ on the product MDP of an MDP and an LDBA associated with φ, then the proposed algorithm will find one such policy**. However, if we construct the product MDP $M^{\otimes}$ of an MDP M and a raw tLDGBA corresponding to a given LTL formula φ and use the reward function defined as the accepting frontier function, a positional policy satisfying φ may not be exist on $M^{\otimes}$ depending on M and φ. We showed an example in which there is no positional policy satisfying φ on $M^{\otimes}$ when using the corresponding raw tLDGBA in our manuscript. In the example, the state of the tLDGBA is always x_0 while the agent does not move to states s_2, s_3, s_5, and s_6 on the original MDP M. Thus, the state of the product MDP is always (s_4, x_0) while the agent stays in s_4 on the original MDP M. Therefore, the method in [2] may not synthesize policies satisfying LTL formulas depending on the setting of MDPs or LTL formulas. Even though we use state-based LDGBA, there is a small counterexample of an MDP M and an LTL formula φ such that there exists no positional policy satisfying φ on the corresponding product MDP $M^{\otimes}$. We show the counterexample as follows. We consider an MDP shown in Fig. 1, the LTL formula $\varphi = \mathbf{GF}a \wedge \mathbf{GF}b$, and the corresponding state-based LDGBA shown in Fig. 2. The acceptance condition of the LDGBA is $\mathcal{F} = \{F_1, F_2\}$, where $F_1 = \{x_1\}$ and $F_2 = \{x_2\}$, which have been removed the accepting state transitioned by the observation $a \wedge b$, respectively. The initial state of the corresponding MDP $M^{\otimes}$ is (s_0, x_0). Let the initial action be $left$. The state of $M^{\otimes}$ is transitioned to (s_1, x_1) from (s_0, x_0) by action $left$. Then action *right* performed at the state (s_1, x_1) and the state is transitioned to (s_0, x_0). Subsequently, the agent have to take the action *right* at (s_0, x_0) eventually to visit all accepting sets F_1 and F_2. However, the action selection is not positional.

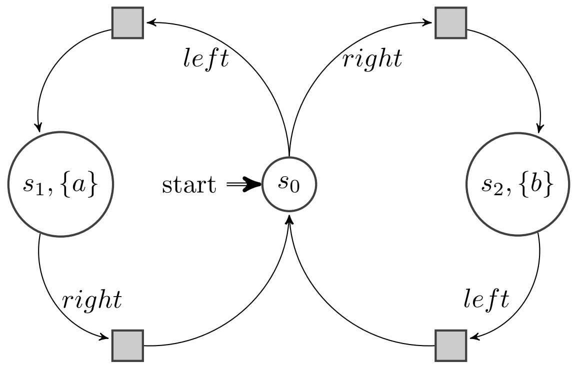

Figure 1: The MDP with three states s_0, s_1, and s_2 and two actions $left$ and $right$. s_1 and s_2 are labeled with $\{a\}$ and $\{b\}$, respectively. The initial sate is s_0.

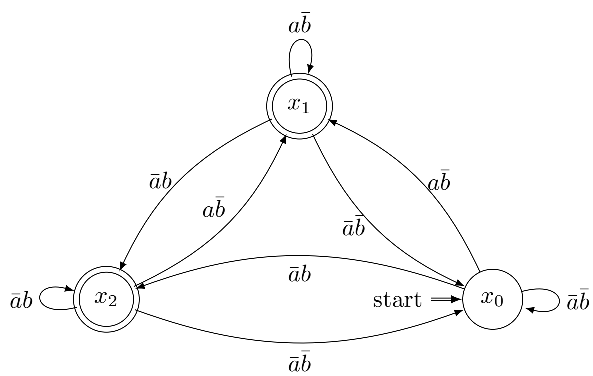

Figure 2: The state-based LDBA corresponding the LTL formula $\varphi = \mathbf{GF}a \wedge \mathbf{GF}b$. The accepting states are represented as double circles. The initial state is x_0.

Review Point 2.3. The main benefit of using a LDBA is that its state space is smaller than alternative automata such Rabin Automata. However, here due to state augmentation (Definition 8) this advantage is lost. The authors need to motivate the use of the LDBA in this paper and also report the size of the state-space of the product MDP and compare it to [14]. The latter is important for the scalability of the proposed algorithm.

Reply. As you pointed out, by the augmentation, the state space of an augmented tLDGBA corresponding to an LTL formula φ is larger than a tLDBA corresponding to the same formula and its size is about $\frac{2^n-1}{n}$ times, where n is the number of all accepting sets of the original tLDGBA. We added a remark

(Remark 1) explaining this point to our manuscript. The reason we use LDBAs instead of other automata is mainly described in [1]. It is known that deterministic Rabin automata (DRA) and non-deterministic Büchi automata (NBA) can recognize all of the ω-regular language. However, there is a counterexample of an MDP M and an LTL formula φ with Rabin index 2, which is mentioned in [1], such that, although there is a positional policy satisfying φ with probability 1 on $M^{\otimes}$ of M and the DRA, optimal policy obtained from any reward based on the acceptance condition of the DRA do not satisfy the LTL formula with probability 1. This is because the reward function is defined for each acceptance pair of the acceptance condition of the DRA, namely the counterexample is due to that only one acceptance pair of the DRA is considered in one learning. Further, LDBAs are not only as expressive as NBAs but also the number of non-deterministic transitions are much less than NBAs.

Review Point 2.4. Does maximizing the collection of the proposed rewards implies maximization of the satisfaction probability? In other words, does the proposed algorithm find a feasible or the optimal solution.

Reply. No, it is. It does not generally hold in our problem settings that maximizing expected discounted reward implies maximizing the satisfaction probability. We revised the conclusion in our manuscript to clarify that maximizing the satisfaction probability is one of future works.

Review Point 2.5. The definition of the labeling function (page 2) is unusual. Typically, observations are assigned to states and not to transitions.

Reply.

Review Point 2.6. In Definition 2, $last(\rho)$ is not defined anywhere in the text.

Reply. We defined $last(\rho)$ in the second paragraph of Definition 1.

3 Reply to Reviewer 7 (20009)

Review Point 3.1. The probability function in Def. 9 can be not well-defined for non-deterministic transitions. Consider a simple example, say (x_1, l, x'_1) and (x_1, l, x'_2) are both in delta. $P(s'|s,a) = 1$ with $l = L((s,a,s'))$ will have two outgoing transitions labeled by the same action a, and all with probability one. The sum of probability given action a on state (s, x_1) will be 2. Will this study only consider deterministic transitions?

Reply. We added the explanation of ε-transitions in a tLDGBA to the definition of the limit-determinism. Then, we revised Definition 8 to clarify how we handle ε-transitions in an augmented tLDGBA.

Review Point 3.2. The construction of augmented automaton in Def. 8 is not well-motivated. What is the intuition behind this construction? Further, there should be a formal proof that the two automata accepting the same language. Examples in section IV helps a lot. It would be useful to have a small example to illustrate the construction.

Reply. The intuition behind the construction is that we think the method in [2] may not have positional policy satisfying a given LTL formula on the corresponding MDP. This is because that the original LDGBAs have less memory capacity than LDBAs and the accepting frontier function is memoryless. We believe it is expanded that the class of positional policies satisfying a given LTL formula on the corresponding product MDP by our augmentation. We added two remarks that explain the reason we used LDGBAs instead of other automata including LDBAs and that if we use original LDGBAs and the reward function defined as the accepting frontier function, there exists an example such that there exists no positional policy satisfying a given LTL formula on the corresponding product MDP. We have showed the example in section IV that the method in [2] cannot synthesize any positional policy satisfying the given LTL formula on the corresponding product MDP.

Due to lack of space in our manuscript, we gave up to add the formal proof to the manuscript. So we proof the two automata accepts the same language only in the reply letter.

Theorem 1. *A tLDGBA and its augmentation accept the same language.*

Proof. Let B denote a tLDGBA and $\bar{B}$ denote its augmentation. Let a infinite word r be accepted by B. Thus, $inf(r) \cap F_j \neq \emptyset$ holds for any accepting set F_j of B, where $inf(r)$ is the set of transitions that occur infnitely often in the word r. In other words, the infinite accepting transitions of all accepting sets of B are in r regardless of its order. Then, we consider how the word r is accepted by $\bar{B}$. By the definition of the acceptance condition of $\bar{B}$, two or more visits to an accepting set F_j does not contribute to the acceptance by $\bar{B}$ until all accepting sets of B are visited after the latest visits to them of B. For a current binary-valued vector v, if a current transition (x, σ, x') is in an accepting set that have been visited once or more after the latest visits to all accepting sets of B, $Max(v, visitf((x, \sigma, x'))) = v$ holds, where $visitf((x, \sigma, x'))_j$ represents the j-th element of $visitf((x, \sigma, x'))$. This is because the j-th element of v is already 1 and $visitf((x, \sigma, x'))$ returns a vector such that the j-th element is 1 and the rest of elements are 0. Therefore, by the definition of the acceptance condition of $\bar{B}$, the transition is not in any accepting sets of $\bar{B}$. While, if the current transition (x, σ, x') is first visit to an accepting set F_j of B after the latest visits to all accepting sets of B, namely the j-th element of the current binary-valued vector is 0, then $Max(v, visitf((x, \sigma, x')))$ returns the vector such that the j-th element is 1 and the rest of elements are same as v. Therefore, by the definition of the acceptance condition of $\bar{B}$, the transition is in the j-th accepting set $\bar{F}_j$ of $\bar{B}$. When all accepting sets of B have visited at the transition $(\hat{x}, \hat{\sigma}, \hat{x}')$, we have $Max(v, visitf(\hat{x}, \hat{\sigma}, \hat{x}'))) = \mathbf{1}$ and $reset(Max(v, visitf((\hat{x}, \hat{\sigma}, \hat{x}')))) = \mathbf{0}$. Subsequently, the same processing is performed on the word r, so that r is accepted by $\bar{B}$. Therefore, the language accepted by B are also accepted by $\bar{B}$.

Let r' denote the word accepted by $\bar{B}$. The word r' visits all accepting sets of $\bar{B}$ infinitely often. Thus, by the definition of the acceptance condition of $\bar{B}$, obviously r' visits all accepting sets of B infinitely often. Therefore, the language accepted by $\bar{B}$ are also accepted by B. $\square$

Review Point 3.3. The notations are too complicated and can be simplified.

Reply.

Review Point 3.4. Def. 7 is bit complicated in writing, the definition in ref.[13] is much clearer. Further, the statement the transitions in each part are deterministic ·· this claim is not suggested in [13]. In fact, in the original definition, the state space partitions to the nondeterministic part and deterministic part. The transitions in the nondeterministic part can be nondeterministic.

Reply. We revised the definition of limit-determinism and employed the definition like [5]. Then we explain that the transitions in initial part are deterministic because of its construction.

4 Reply to Reviewer 8 (20011)

Review Point 4.1. The paper is well-written and organized well. There are lot of symbols with subscripts, superscripts in the paper and it might be helpful to include a paragraph that describes the notation.

Reply.

Review Point 4.2. How is the reward function defined in the example for the method in [2]? Can the two rewards be compared?

Reply. We defined the reward function in the example for the method in [2] as the accepting frontier function defined in [2] and the immediate reward is the same value as the reward for our proposed method. If an optimal policy on the product MDP were satisfying the LTL formula φ, the agent could visit s_0, s_8 infinitely often and would never visit s_2, s_3, s_5, and s_6, and would accumulate rewards just like the agent under the our proposed method. Therefore, the two rewards can be compared.

Review Point 4.3. How does the proposed method compare in terms of computational complexity with other methods? This would be helpful information for the readers especially since the proposed method depends on augmenting the LDBA.

Reply. We interpret the computational complexity as the size of the state space of an automaton. In general, when constructing a transition-based Büchi automaton (tBA) from a transition-based generalized Büchi automaton (tGBA), the order of visits to accepting sets of the tGBA is fixed. Consequently, the reward based on the acceptance condition of the tBA tends to be sparse and the sparsity is critical against RL-based control policy synthesis problems. The augmentation of tGBA relaxes the sparsity since the augmented tGBA has all of the order of visits to all accepting sets of the original tGBA. The size of the state space of the augmented tGBA is about $\frac{2^n-1}{n}$ times larger than the tBA, however, the ratio of the number of accepting transitions to the number of all transitions of the augmented tGBA is much greater than the tBA.

References

[1] E. M. Hahn, M. Perez, S. Schewe, F. Somenzi, A. Triverdi, and D. Wojtczak, "Omega-regular objective in model-free reinforcement learning," *Lecture Notes in Computer Science*, no. 11427, pp. 395–412, 2019.

[2] M. Hasanbeig, A. Abate, and D. Kroening, "Logically-constrained reinforcement learning," *arXiv:1801.08099v8*, Feb. 2019.

[3] M. Hasanbeig, Y. Kantaros, A. Abate, D. Kroening, G. J. Pappas, and I. Lee, "Reinforcement learning for temporal logic control synthesis with probabilistic satisfaction guarantee," *arXiv:1909.05304v1*, 2019.

[4] A. K. Bozkurt, Y. Wang, M. Zavlanos, and M. Pajic, "Control synthesis from linear temporal logic specifications using model-free reinforcement learning," *arXiv:1909.07299*, 2019.

[5] S. Sickert, J. Esparaza, S. Jaax, and J. Křetìnský, "Limit-deterministic Büchi automata for linear temporal logic," in *International Conference on Computer Aided Verification*, 2016, pp. 312-332.