Author's Reply.

Ryohei Oura, Ami Sakakibara, and Toshimitsu Ushio

February 6, 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. If it it no need to distinguish transition-besed from state-based, we represent limit-deterministic Büchi automata and non-generalized one as LDGBA and LDBA, respectively.

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 tLDGBA $\bar{B}_\varphi$ 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, your 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 tLDGBA 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 a sufficiently large discount factor if there exists a positional policy on $M^\otimes$ satisfying φ. 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 one of reason we use LDGBAs is that an LDGBA is not only as expressive as NBAs but also the number of non-deterministic transitions are much less than a non-deterministic Büchi automaton (NBA). Advantages of LD(G)BAs over other automata are explained 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, such that, although there is a positional policy satisfying φ with probability 1 on $M^\otimes$ of M and the DRA, an optimal policy obtained from any reward based on the acceptance condition of the DRA does 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.

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. We added the explanation of some notations after the introduction.

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 obtained policies. This notion also enables us more general problem settings, e.g., to find a policy that maximizing 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?

Also, there are several recent papers addressing similar problems that need to be discussed:

Li, Xiao, et al. "A formal methods approach to interpretable reinforcement learning for robotic planning." Science Robotics 4.37 (2019).

Gao, Qitong, et al. "Reduced variance deep reinforcement learning with temporal logic specifications." Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems. ACM, 2019.

Reply. Yes, we mean that the proposed method is more sample efficient. By Definitions 8 and 10 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.

On the other hand, the reward function defined by the accepting frontier function [2], however, has no memory of previous visits to the accepting sets of the original LDGBA although they construct the product MDP from the original LDGBA. Therefore, there is an example of an MDP M and an LDGBA converted from an LTL formula φ that there exists no positional policy satisfying φ on the product MDP of M and a tLDGBA corresponding to φ. In order to obtain a positional policy φ, we may have to a redundant LDGBA heuristically. However, the heuristic redundancy can cause the sparsity of rewards and a search space larger than necessary, and so on.

We added the discussion of the two papers in the introduction. However, due to the limitation of the manuscript, we can only make a brief discussion in the manuscript. So we discuss again about the two papers in the letter.

- Li, Xiao, et al. "A formal methods approach to interpretable reinforcement learning for robotic planning." Science Robotics 4.37 (2019).

- Gao, Qitong, et al. "Reduced variance deep reinforcement learning with temporal logic specifications." Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems. ACM, 2019.

  In the paper, they tackle to make a low variance deep reinforcement learning method for LTL tasks. They convert an given LTL formula to a deterministic Rabin automaton. Then, they take a product from an MDP and the Rabin automaton. The reward function is based on the acceptance condition of a deterministic Rabin automaton and assigns a reward according to the Rabin state. The key idea of their proposed method is to convert deep Q-learning problem minimizing the error to a nonconvex mini-max optimization problem. Consequently, they achieve a state-action value function than a normal stochastic gradient descent method.

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. We revised the statement of the results to clarify that (i) the proposed method is more sample efficient than using LDBAs and (ii) the method in [2] cannot synthesize the positional policy satisfying the LTL formula φ on the product MDP in the example and we have to make a redundant LDGBA in order to obtain a positional policy satisfying φ. 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 B_{φ} corresponding to a given LTL formula φ and use the reward function defined by the accepting frontier function, then a positional policy satisfying φ may not be exist on $M^{\otimes}$ depending on M and B_{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 in Fig. 1 is always x_0 while the agent does not move to bad states s_2, s_3, s_5, and s_6 on the original MDP M. Thus, unless the bad states is visited, the product state is always (s_4, x_0) while the agent is in s_4 of the original MDP M. That is, in order to obtain a positional policy satisfying a given LTL formula φ with the method in [2, 3], we may have to make a redundant LDGBA associated with φ heuristically. However, the heuristic redundancy can cause a sparsity of rewards and a search space larger than necessary, and so on. Even if 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 Figure 1, the LTL formula $\varphi = \mathbf{GF}a \wedge \mathbf{GF}b$, and the corresponding non-redundant state-based LDGBA shown in Figure 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 the action $left$. Then the action $right$ is 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 of $M^{\otimes}$ corresponding to F_1 and F_2. However, the action selection is not positional.

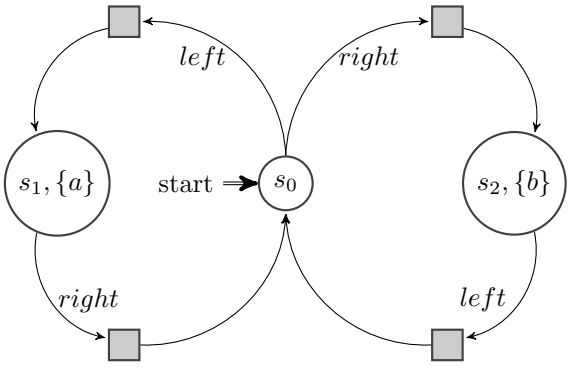

Figure 1: The MDP with three states s_0, s_1, and s_2 and two actions $left$ and $right$. s_1 and s_2 are labeled with $\{a\}$ and $\{b\}$, respectively. The initial sate is s_0.

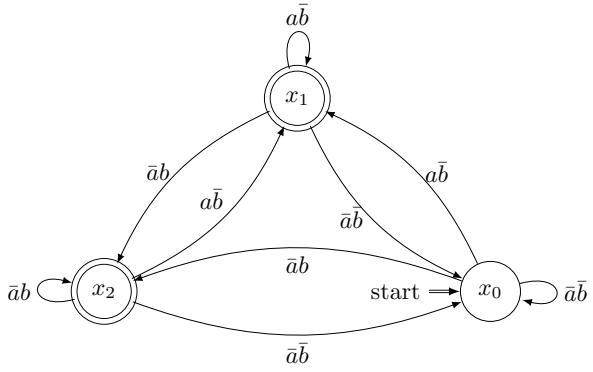

Figure 2: The state-based LDBA corresponding the LTL formula $\varphi = \mathbf{GF}a \wedge \mathbf{GF}b$. The accepting state corresponding to the observation $a \wedge b$ are removed. The accepting states are represented as double circles. The initial state is x_0.

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. The one of reason we use LDGBAs is that an LDGBA is not only as expressive as a non-deterministic Büchi automaton (NBA) but also the number of non-deterministic transitions are much less than a NBA. As you pointed out, by Definition 8, the state space of an augmented tLDGBA from a tLDGBA converted from an LTL formula φ is $\frac{2^n-1}{n}$ times larger than a tLDBA constructed from the tLDGBA, where n is the number of all accepting sets of the original tLDGBA. However, the number of accepting transitions to all transitions in the augmented tGBA is much greater than the tBA. Therefore, our proposed method is expected to be more sample efficient than using tLDBAs. We added a remark (Remark 1) explaining this point to our manuscript. Advantages of LD(G)BAs over other automata is explained in [1]. There is a counterexample of an MDP M and a deterministic Rabin automaton converted from an LTL formula φ, such that, although there is a positional policy satisfying φ with probability 1 on $M^{\otimes}$ of M and a DRA converted from φ, any optimal policy obtained from any reward based on the acceptance condition of the DRA does not satisfy the LTL formula with probability 1. This is because the reward functions are 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.

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 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. Some existing works employ the definition of the labeling function that assigns an observation to a transition in a controlled system [6, 7]. Then we define the labeling function like them.

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 9 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 two intuition behind the construction are as forrows. (i) When constructing a Büchi automaton (BA) from a generalized Büchi automaton (GBA), the order of visits to accepting sets of the GBA is fixed, and thus the conversion cause the sparsity of rewards. (ii) The method in [2, 3] may not have a positional policy satisfying a given LTL formula on the corresponding product MDP even if there exists a policy satisfying the LTL formula on the original MDP. For first intuition, we added a remark (Remark 1) that explains the sparsity of rewards from the conversion. For second intuition, we showed an example in which the method in [2, 3] has no policy satisfying the given LTL formula with non-redundant tLDGBA.

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

Notations

- $\mathcal{L}(A) \subseteq \Sigma^\omega$: the accepted language of an automaton A with the alphabet Σ, namely, the set of all infinite words accepted by A.

Proposition Let $B = (X, x_{init}, \Sigma, \delta, \mathcal{F})$ and $\bar{B} = (\bar{X}, \bar{x}_{init}, \bar{\Sigma}, \bar{\delta}, \bar{\mathcal{F}})$ be an arbitrary tLDGBA and its augmentation, respectively. Then, we have $\mathcal{L}(B) = \mathcal{L}(\bar{B})$.

Proof

$\subseteq$: Consider any $w = \sigma_0\sigma_1 \ldots \in \mathcal{L}(B)$. Then, there exists a run $r = x_0\sigma_0x_1\sigma_1x_2 \ldots \in X(\Sigma X)^\omega$ of B such that $x_0 = x_{init}$ and $inf(r) \cap F_j \neq \emptyset$ for each $F_j \in \mathcal{F}$. Recall that $\Sigma = \bar{\Sigma}$. For the run r, we construct a sequence $\bar{r} = \bar{x}_0\bar{\sigma}_0\bar{x}_1\bar{\sigma}_1\bar{x}_2 \ldots \in \bar{X}(\bar{\Sigma}\bar{X})^\omega$ satisfying $\bar{x}_i = (x_i, v_i)$ and $\bar{\sigma}_i = \sigma_i$ for any $i \in \mathbb{N}$, where

$$v_0 = \mathbf{0} \text{ and } \forall i \in \mathbb{N},\ v_{i+1} = reset\Big(Max\big(v_i, visitf((x_i, \bar{\sigma}_i, x_{i+1}))\big)\Big).$$

Clearly from the construction, we have $(\bar{x}_i, \bar{\sigma}_i, \bar{x}_{i+1}) \in \bar{\delta}$ for any $i \in \mathbb{N}$. Thus, $\bar{r}$ is a run of $\bar{B}$ starting from $\bar{x}_0 = (x_{init}, \mathbf{0}) = \bar{x}_{init}$.

We now show that $inf(\bar{r}) \cap \bar{F}_j \neq \emptyset$ for each $\bar{F}_j \in \bar{\mathcal{F}}$. Since $inf(r) \cap F_j \neq \emptyset$ for each $F_j \in \mathcal{F}$, we have

$$\forall j \in \{1, \ldots, n\},\ inf(\bar{r}) \cap \{\big((x, v), \bar{\sigma}, (x', v')\big) \in \bar{\delta} : visitf((x, \bar{\sigma}, x'))_j = 1\} \neq \emptyset.$$

By the construction of $\bar{r}$, therefore, there are infinitely many indices $l \in \mathbb{N}$ with $v_l = \mathbf{0}$. Let $l_1, l_2 \in \mathbb{N}$ be arbitrary nonnegative integers such that $l_1 < l_2$, $v_{l_1} = v_{l_2} = \mathbf{0}$, and $v_{l'} \neq \mathbf{0}$ for any $l' \in \{l_1 + 1, \ldots, l_2 - 1\}$. Then,

$$\forall j \in \{1, \ldots, n\},\ \exists k \in \{l_1, l_1{+}1, \ldots, l_2{-}1\},\ (x_k, \sigma_k, x_{k+1}) \in F_j \wedge (v_k)_j = 0,$$

where $(v_k)_j$ is the j-th element of v_k. Hence, we have $inf(\bar{r}) \cap \bar{F}_j \neq \emptyset$ for each $\bar{F}_j \in \bar{\mathcal{F}}$, which implies $w \in \mathcal{L}(\bar{B})$.

$\supseteq$: Consider any $\bar{w} \in \bar{\sigma}_0\bar{\sigma}_1 \ldots \in \mathcal{L}(\bar{B})$. Then, there exists a run $\bar{r} = \bar{x}_0\bar{\sigma}_0\bar{x}_1\bar{\sigma}_1\bar{x}_2 \ldots \in \bar{X}(\bar{\Sigma}\bar{X})^\omega$ of $\bar{B}$ such that $\bar{x}_0 = \bar{x}_{init}$ and $inf(\bar{r}) \cap \bar{F}_j \neq \emptyset$ for each $\bar{F}_j \in \bar{\mathcal{F}}$, i.e.,

$$\forall j \in \{1, \ldots, n\},\ \forall k \in \mathbb{N},\ \exists l \geq k,\ ([\![\bar{x}_l]\!]_X, \bar{\sigma}_l, [\![\bar{x}_{l+1}]\!]_X) \in F_j \wedge (\bar{v}_l)_j = 0, \tag{1}$$

where $[\![(x, v)]\!]_X = x$ for each $(x, v) \in \bar{X}$. For the run $\bar{r}$, we construct a sequence $r = x_0\sigma_0x_1\sigma_1x_2 \ldots \in X(\Sigma X)^\omega$ such that $x_i = [\![\bar{x}_i]\!]_X$ and $\sigma_i = \bar{\sigma}_i$ for any $i \in \mathbb{N}$. It is clear that r is a run of B starting from $x_0 = x_{init}$. It holds by Eq. (1) that $inf(r) \cap F_j \neq \emptyset$ for each $F_j \in \mathcal{F}$, which implies $\bar{w} \in \mathcal{L}(B)$.

We showed an example of tLDGBA and its augmentation after Definition 8.

Review Point 3.3. In the proof of Lemma 1, the last sentence is unclear. The goal is to show that either an agent receives no reward from visiting recurrent class, or the agent has to visit all recurrent class for which the acceptance condition are satisfied · It is not clear why it is not possible that a policy only visits some subset of acceptance set but not all. This may due to the lack of explanation to the def.8.

Reply. We consider the unclarity of the proof of Lemma 1 is due to the lack of the explanation of Definition 8. So we added a statement with regard to the acceptance condition of an augmented automaton after its definition.

Review Point 3.4. The construction of augmented automaton is similar to the accepting frontier function in the following paper, presented at CDC this year: Hosein et al. mainly, the same problem was studied too. The authors should highlight the difference and their unique contribution comparing to existing work. Reinforcement Learning for Temporal Logic Control Synthesis with Probabilistic Satisfaction Guarantees https://arxiv.org/pdf/1909.05304.pdf

Reply. We added a remark (Remark 2) to clarify the comparison between our proposed method and the method in [2, 3].

Review Point 3.5. The comparison with [14] is unclear. If both methods generate the optimal policies, then it is not clear why [14] does not perform correctly in this example.

Reply. We revised the statement of the results. The result for the method in [2, 3] is because it is impossible the transitions labeled with $\{a\}$ and $\{b\}$ occur from s_4 infinitely often by any positional policy with B_φ shown in Fig. 1. In detail, the state of the B_φ is always x_0 while the agent does not move to bad states s_2, s_3, s_5, and s_6. Thus, unless the bad states are visited, the product state is always (s_4, x_0) while the agent is in s_4. Thus, the agent cannot visit both of s_0 and s_8 by a deterministic action selection at s_4. While our proposed method can recognize the previous visits as a state. Thus, our proposed method can synthesize a positional policy satisfying φ on the product MDP, while the method in [2, 3] cannot. That is, in order to obtain a positional policy satisfying a given LTL formula φ with the method in [2, 3], we may have to make a redundant LDGBA associated with φ heuristically. However, the heuristic redundancy can cause a sparsity of rewards and a search space larger than necessary, and so on.

Review Point 3.6. The notations are too complicated and can be simplified.

Reply. For simplicity, we omitted the superscripts $^\otimes$ with product states in the proof of Theorem 1.

Review Point 3.7. 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. For simplicity, we omitted the superscripts $^{\otimes}$ with product states in the proof of Theorem 1.

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.

[6] G. H. Mealy, "A method for synthesizing sequential circuits." *The Bell System Technical Journal* 34.5 (1955): 1045-1079.

[7] A. Camacho, R. T. Icarte, T. Q. Klassen, R. Valenzano, "LTL and beyond: Formal languages for reward function specification in reinforcement learning." *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*. 2019.