Easy Complementation of History-Deterministic Büchi Automata

Bader Abu Radi[0000−0001−8138−9406], Orna Kupferman[0000−0003−4699−6117], and Ofer Leshkowitz[0000−0001−9225−2325]

School of Engineering and Computer Science, Hebrew University, Jerusalem, Israel

Abstract. A *history deterministic* (HD) automaton can successfully resolve its nondeterministic choices in a way that only depends on the past. Formally, an HD automaton has a strategy that maps each finite word to the transition to be taken after the word is read, and following this strategy results in accepting all the words in the language of the automaton. HD automata can replace deterministic automata in several applications, most notably reactive synthesis, and they attract a lot of interest in the research community.

In 2015, Kuperberg and Skrzypczak proved that an HD Büchi automaton $\mathcal{A}$ can be determinized with a quadratic blow-up in the state space. The suggested construction also implicitly induces an HD co-Büchi automaton of linear size that complements $\mathcal{A}$. The construction is based on a series of transformations on an HD strategy for $\mathcal{A}$, which has two drawbacks. First, the construction is conceptually hard to understand. Second, since the state space of the HD strategy is exponential, so is the runtime of a procedure implementing the construction.

In this paper, we describe a direct complementation procedure for HD Büchi automata. Our procedure is based on removal of transitions from $\mathcal{A}$ itself, which can be done in polynomial time. It results in an HD Büchi automaton equivalent to $\mathcal{A}$ on top of which we define a complementing HD co-Büchi automaton of linear size. A deterministic equivalent automaton of size quadratic in $\mathcal{A}$ can then be defined in polynomial time too. We also prove a quadratic lower bound on the size of an HD strategy for HD Büchi automata.

1 Introduction

Nondeterminism allows a computing machine to examine several possible actions simultaneously [19]. In the setting of automata on finite words, nondeterminism does not add to the expressive power of deterministic automata, yet enables the definition of exponentially more succinct automata. In the setting of *automata on infinite words* [6], nondeterminism is more involved. Acceptance in such automata is determined according to the set of transitions that are traversed infinitely often along the run. For example, in *Büchi* automata (tNBW and tDBW, for nondeterministic and deterministic Büchi word automata, respectively), the acceptance condition is a subset α of transitions, and a run is

accepting iff it visits α infinitely often.[1] Consequently, the *subset construction*, which is useful for determinization of automata on finite words, does not work for automata on infinite words, as it does not maintain information on the history of runs beyond the state they have reached. In fact, NBWs are strictly more expressive than DBWs [17].

Automata on infinite objects are used in *specifications*, *verification*, and *synthesis* of reactive systems [21,13]. In some applications, such as verification, algorithms can be based on nondeterministic automata, whereas in other applications, such as synthesis and control, algorithms are based on deterministic automata, and thus involve a complicated determinization construction [20] or acrobatics for circumventing determinization [16,8]. Essentially, the problem with nondeterminism is that it amounts to guessing the future, and in many applications, the use of nondetermnistic automata means that the same guess is applied to all futures, which is too restrictive.

In [9], Henzinger and Piterman introduced *history deterministic* (HD) automata, which capture this difficulty in a very clean way. A nondeterministic automaton is HD if it can to resolve its nondeterministic choices in a way that only depends on the past.[2] Formally, a nondeterministic automaton $\mathcal{A}$ over an alphabet Σ is HD if there is a strategy g that maps each finite word $u \in \Sigma^*$ to the transition to be taken after u is read; and following g results in accepting all the words in the language of $\mathcal{A}$. Note that a state q of $\mathcal{A}$ may be reachable via different words, and g may suggest different transitions from q after different words are read.

The simplest class of HD automata are deterministic ones. Another simple class consists of nondeterministic automata that are *determinizable by pruning* (DBP); that is, ones that just add transitions on top of a deterministic automaton. Clearly, DBP automata are HD, as witnessed by a strategy that follows the embodied deterministic automaton. In fact, it is not hard to see that HD nondeterministic automata on finite words are always DBP. Moving to automata on infinite words, the picture is different. On the one hand, in terms of expressive power, HD automata are not more expressive than deterministic ones [15,18]. On the other hand, HD automata need not be DBP [3], and they constitute an interesting and intriguing class of automata, many of whose properties are still unknown [14,5].

In [12], Kuperberg and Skrzypczak study determinization of HD automata. They focus on Büchi auotmata and their dual *co-Büchi* automata (denoted tDCW and tNCW, for the deterministic and nondeterministic classes). For HD-

[1] Traditionally, the study of automata on infinite words assume *state-based* acceptance conditions, which specify the set of states that should be visited infinitely or finitely often. Here, we assume *transitions-based* acceptance, which turns out to be cleaner in our study.

[2] The notion used in [9] is *good for games* (GFG) automata, as they address the difficulty described above in the context of games played on top of a nondeterministic automaton. As it turns out, the property of being good for games varies in different settings, and history determinism is useful for applications beyond games [15,7], and so we opt for using the term HD.

tNCWs, the picture is well understood: Every HD-tNCW can be determinized with an exponential blow-up, and this bound is tight [12]. For Büchi automata, Kuperberg and Skrzypczak proved that an HD-tNBW $\mathcal{A}$ can be determinized with a quadratic blow-up in the state space, and no lower bound is known. The suggested determinization construction also implicitly induces an HD-tNCW of linear size that complements $\mathcal{A}$.

Essentially, the construction in [12] starts with computing an HD strategy g for $\mathcal{A}$ that is based on the subset-construction, and is therefore exponential. It then considers a *parity game* defined on top of the product of $\mathcal{A}$ and g. Using ranks that are defined on top of the game, the construction identifies properties of $\mathcal{A}$ and g that are used in order to modify them while preserving $\mathcal{A}$'s language and HDness.

While the construction cleverly solves the open problem of polynomial determinization of HD-tNBWs, it has two drawbacks. First, its implementation is complex: it is based on a product with an exponential HD strategy and thus requires exponential time, and it involves optimal strategies in parity games. Second, the construction involves ranks obtained by analyzing a game played on top of both $\mathcal{A}$ and g. This makes the construction very hard to understand. In addition, it is difficult to gain a conceptual understanding of HDness from it.

In this paper we suggest an alternative complementation construction for HD-tNBWs, which addresses the above two drawbacks. Our construction is direct, and can be implemented in time polynomial in $\mathcal{A}$. We show that a complementing automaton can be defined on top of the state space of $\mathcal{A}$, after removing from it transitions that are not essential for its language or its HDness. We prove the correctness of our construction by analyzing the structure of a witness HD strategy. Thus, as in [12], we analyze HD strategies, yet we are able to avoid ranks and games, and the analysis is used only for proving correctness – the construction itself is independent of such a strategy. This makes the proof cleaner, shorter, and easier to understand.

Our complementation construction also yields a new determinization construction: once we complement $\mathcal{A}$ to an HD-tNCW, a tDBW equivalent to $\mathcal{A}$ can be defined on top of the product of $\mathcal{A}$ and its complement [3]. Thus, in polynomial time we also determinize $\mathcal{A}$ with a quadratic blow-up. Our construction also implies a quadratic upper bound for the size of HD strategies. We show that the bound is tight. That is, we show that for every $n \geq 1$, there is an HD-tNBW $\mathcal{A}_n$ (in fact, even one that is *deterministic in the limit*) such that $\mathcal{A}_n$ has $O(n)$ states and every HD strategy for $\mathcal{A}_n$ needs at least n^2 states. Note that this does not imply a quadratic lower bound for determinization of HD-tNBWs. Indeed, it only implies that determinization constructions that are based on pruning strategies are at least quadratic, suggesting that a linear determinization construction, if exists, must involve redirectioning of transitions.

We believe that our results contribute not only to efficient complementation and determinization constructions for HD Büchi automata, but also to better understanding of history determinism.

2 Preliminaries

For a finite nonempty alphabet Σ, an infinite *word* $w = \sigma_1 \cdot \sigma_2 \cdots \in \Sigma^\omega$ is an infinite sequence of letters from Σ. For positions $1 \leq i \leq j$, we use $w[i,j]$ to denote the finite infix $\sigma_i \cdot \sigma_{i+1} \cdots \sigma_j$ of w, and use $w[i,\infty]$ to denote the infinite suffix $\sigma_i \cdot \sigma_{i+1} \cdots$. A *language* $L \subseteq \Sigma^\omega$ is a set of infinite words. We use $\overline{L}$ to denote the complement of L, thus $\overline{L} = \Sigma^\omega \setminus L$. We sometimes refer also to finite words, namely elements in Σ^*, and denote the empty word by ϵ.

A *nondeterministic automaton* on infinite words is $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$, where Σ is an alphabet, Q is a finite set of *states*, $q_0 \in Q$ is an *initial state*, $\delta : Q \times \Sigma \to 2^Q \setminus \emptyset$ is a *transition function*, and α is an *acceptance condition*, to be defined below. We extend δ to sets of states and finite words in the expected way. Thus, $\delta(S,u)$ is the set of states that $\mathcal{A}$ may reach when it reads the word $u \in \Sigma^*$ from some state in $S \in 2^Q$. Formally, $\delta : 2^Q \times \Sigma^* \to 2^Q$ is such that for every $S \in 2^Q$, finite word $u \in \Sigma^*$, and letter $\sigma \in \Sigma$, we have that $\delta(S,\epsilon) = S$, $\delta(S,\sigma) = \bigcup_{s \in S} \delta(s,\sigma)$, and $\delta(S, u \cdot \sigma) = \delta(\delta(S,u),\sigma)$. Note that δ is *full*, thus $|\delta(q,\sigma)| \geq 1$ for every state $q \in Q$ and letter $\sigma \in \Sigma$. If $|\delta(q,\sigma)| = 1$ for every state $q \in Q$ and letter $\sigma \in \Sigma$, then $\mathcal{A}$ is *deterministic*. In a deterministic automaton $\mathcal{A}$, we view δ and its extension to words as a function $\delta : Q \times \Sigma^* \to Q$. We sometimes refer to the transition function δ as a transition relation $\Delta \subseteq Q \times \Sigma \times Q$, where for every two states $q, s \in Q$ and letter $\sigma \in \Sigma$, we have that $\langle q, \sigma, s \rangle \in \Delta$ iff $s \in \delta(q,\sigma)$.

An *run* of $\mathcal{A}$ on an infinite word $w = \sigma_1 \cdot \sigma_2 \cdots \in \Sigma^\omega$, is an infinite sequence $r \in \Delta^\omega$ of successive transitions that are traversed while reading w. Thus, $r = \langle r_0, \sigma_1, r_1 \rangle, \langle r_1, \sigma_2, r_2 \rangle, \langle r_2, \sigma_3, r_3 \rangle, \ldots$ is such that $r_0 = q_0$, and for all $i \geq 0$, we have that $r_{i+1} \in \delta(r_i, \sigma_{i+1})$. We sometimes also consider runs of $\mathcal{A}$ on a finite words $w = \sigma_1 \cdot \sigma_2 \cdots \sigma_k \in \Sigma^*$, where $r = \langle r_0, \sigma_1, r_1 \rangle, \langle r_1, \sigma_2, r_2 \rangle, \ldots, \langle r_{k-1}, \sigma_k, r_k \rangle$, and then similarly $r_0 = q_0$, and for all $0 \leq i \leq k-1$, we have that $r_{i+1} \in \delta(r_i, \sigma_{i+1})$. The acceptance condition α determines which runs are "good". We consider here the *Büchi* and *co-Büchi* transition-based acceptance conditions, where $\alpha \subseteq \Delta$ is a subset of transitions. For an infinite run r, let $\mathit{inf}(r) \subseteq \Delta$ be the set of transitions that r traverses infinitely often. Thus, $\mathit{inf}(r) = \{t \in \Delta : t = \langle r_i, \sigma_{i+1}, r_{i+1} \rangle \text{ for infinitely many } i\text{'s}\}$. A run r of a Büchi automaton is *accepting* iff it traverses transitions in α infinitely often, thus $\mathit{inf}(r) \cap \alpha \neq \emptyset$. Dually, a run r of a co-Büchi automaton is accepting iff it traverses transitions in α only finitely often, thus $\mathit{inf}(r) \cap \alpha = \emptyset$. A run that is not accepting is *rejecting*. For finite words the acceptance condition α is a subset of *accepting states*. Thus, $\alpha \subseteq Q$ and a run r is *accepting* if the last state in r belongs to α. Note that as $\mathcal{A}$ is nondeterministic, it may have several runs on a word w. The word w is accepted by $\mathcal{A}$ if there is an accepting run of $\mathcal{A}$ on w. The language of $\mathcal{A}$, denoted $L(\mathcal{A})$, is the set of words that $\mathcal{A}$ accepts.

Two automata are *equivalent* if their languages are equivalent. For a state $q \in Q$, we define $\mathcal{A}^q = \langle \Sigma, Q, q, \delta, \alpha \rangle$. Thus, $\mathcal{A}^q$ is obtained from $\mathcal{A}$ by setting the initial state to q. Then, two states $q, s \in Q$ are *equivalent*, denoted $q \sim_{\mathcal{A}} s$, if $L(\mathcal{A}^q) = L(\mathcal{A}^s)$. When $\mathcal{A}$ is clear from the context, we omit it and write $L(q) = L(s)$, $q \sim s$, etc.

We use tDBW, tDCW, tNBW and tNCW to denote deterministic Büchi, deterministic co-Büchi, nondeterministic Büchi and nondeterministic co-Büchi word automata with a transition-based acceptance condition.

A nondeterministic automaton $\mathcal{A}$ is *semantically deterministic* (*SD*, for short) if different nondeterministic choices lead to equivalent states. Thus, for all states $q \in Q$, and transitions $\langle q, \sigma, s_1\rangle, \langle q, \sigma, s_2\rangle \in \Delta$, it holds that $s_1 \sim s_2$. Consider two states q and s in an SD automaton $\mathcal{A}$. It is easy to see that if $q \sim_{\mathcal{A}} s$, then for every $\sigma \in \Sigma$, $q' \in \delta(q, \sigma)$, and $s' \in \delta(s, \sigma)$, we have that $q' \sim_{\mathcal{A}} s'$. An iterative application of the above property, termed the *SDness property*, implies also that for every word $x \in \Sigma^*$, if $q' \in \delta(q, x)$, and $s' \in \delta(s, x)$, we have that $q' \sim_{\mathcal{A}} s'$. In particular, for all finite words x, all the states in $\delta(q_0, x)$ are equivalent. Intuitively, it means that a run of an SD automaton is allowed to take finitely many 'bad' nondeterministic choices; when reading a word in the language there is always a chance to generate an accepting run.

An automaton $\mathcal{A}$ is *history deterministic* (*HD*, for short) if its nondeterminism can be resolved based on the past, thus on the prefix of the input word read so far. Formally, $\mathcal{A}$ is *HD* if there is a *strategy* $g : \Sigma^* \to Q$ such that the following hold:

1. The strategy g is consistent with the transition function. That is, $g(\epsilon) = q_0$, and for every finite word $u \in \Sigma^*$ and letter $\sigma \in \Sigma$, we have that $\langle g(u), \sigma, g(u \cdot \sigma)\rangle \in \Delta$.
2. Following g causes $\mathcal{A}$ to accept all the words in $L(\mathcal{A})$. That is, for every infinite word $w = \sigma_1 \cdot \sigma_2 \cdots \in \Sigma^\omega$, if $w \in L(\mathcal{A})$, then the run $\langle g(\epsilon), \sigma_1, g(w[1,1])\rangle$, $\langle g(w[1,1]), \sigma_2, g(w[1,2])\rangle, \langle g(w[1,2]), \sigma_2, g(w[1,3])\rangle, \ldots$, which we denote by $g(w)$, is an accepting run of $\mathcal{A}$ on w.

We say that the strategy g is an *HD strategy* for $\mathcal{A}$, and that g *witnesses* $\mathcal{A}$'s HDness. We say that a transition $\langle q, \sigma, s\rangle \in \Delta$ is *used by* g if there is $u \in \Sigma^*$ such that $g(u) = q$ and $g(u \cdot \sigma) = s$. Note that every deterministic automaton is HD.

For an automaton $\mathcal{A}$, we say that a state q of $\mathcal{A}$ is *HD* if $\mathcal{A}^q$ is HD. We use HD-tNBW and HD-tNCW to denote history deterministic tNBWs and tNCWs, respectively.

Consider a directed graph $G = \langle V, E\rangle$. A *strongly connected set* in G (SCS, for short) is a set $C \subseteq V$ such that for every two vertices $v, v' \in C$, there is a (possibly empty) path from v to v'. A SCS is *maximal* if it is maximal with respect to containment, that is, for every non-empty set $C' \subseteq V \setminus C$, it holds that $C \cup C'$ is not a SCS. The *maximal strongly connected sets* are also termed *strongly connected components* (SCCs, for short).

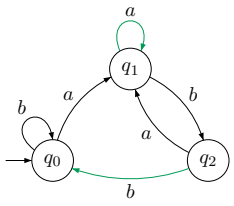
Consider a tNBW $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha\rangle$. Let $G_{\mathcal{A}}^{black} = \langle Q, E_{\mathcal{A}}^{black}\rangle$ be the directed graph with $\langle q, q'\rangle \in E_{\mathcal{A}}^{black}$ iff there is a letter $\sigma \in \Sigma$ such that $\langle q, \sigma, q'\rangle \in \Delta \setminus \alpha$. Thus, $E_{\mathcal{A}}^{black}$ is induced by transitions of $\mathcal{A}$ that are not in α. The *black components* of $\mathcal{A}$, denoted by $\mathcal{B}(\mathcal{A})$, are the SCCs of $G_{\mathcal{A}}^{black}$. For a state $q \in Q$, we denote the black component containing q by $black(q)$.

We say that a tNBW $\mathcal{A}$ is *normal* if transitions between different black components are α transitions. Since a run of $\mathcal{A}$ that traverses infinitely many

transitions between black components does not get stuck in a black component, and is thus accepting, every tNBW can be made normal by adding to α transitions between black components. Note that normalization can be done in linear time and does not change the set of runs nor their classification to accepting and rejecting runs. In particular, normalization maintains HDness. We thus assume that all tNBWs are normal.

We refer to transitions in $\Delta \setminus \alpha$ as *black transitions* and to transitions in α as *green transitions*. Note that in a normal tNBWs, all transitions between black components are green.[3] For a finite or infinite run $r = \langle r_0, \sigma_1, r_1\rangle, \langle r_1, \sigma_2, r_2\rangle, \ldots$, we say that r is black if r is empty or consists of only black transitions. Otherwise, we say that r is green. Thus, r is green if there is $j \geq 0$ such that $\langle r_j, \sigma_j, r_{j+1}\rangle \in \alpha$. Note that $\mathcal{A}$ is normal iff for all states $q \in Q$, black runs from q do not leave $\mathit{black}(q)$. We define the *black language* of q, denoted $L_{black}(\mathcal{A}^q)$, as the set of infinite words w such that there is a black run of $\mathcal{A}^q$ on w.

Example 1. The tDBW $\mathcal{A}$ on the right is not normal. Indeed, $\mathcal{B}(\mathcal{A}) = \{\{q_0\}, \{q_1, q_2\}\}$, and the transition $\langle q_0, a, q_1\rangle$, which connects two black components, is black. Note that the run of $\mathcal{A}$ on the word $w = (ab)^\omega$ does not traverse α, and so $w \in L_{black}(\mathcal{A}^{q_0})$. Normalizing $\mathcal{A}$, the transition $\langle q_0, a, q_1\rangle$ becomes green. Then, the word w is no longer in $L_{black}(\mathcal{A}^{q_0})$, yet the run on it traverses only one green transition, and is thus rejecting.



3 Observations on HD strategies

Recall that each HD automaton has an HD strategy that witnesses its HDness. By [4], each HD automaton has a *finite-state* HD strategy, namely one generated by a *finite-state transducer*. In this section we define *super-tight* HD-tNBWs, which are HD-tNBWs whose HDness is witnessed by a strategy with some helpful properties. We then use this strategy in order to complement HD-tNBWs.

3.1 Tight HD automata

Consider an HD-tNBW $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha\rangle$. We view an HD strategy for $\mathcal{A}$ as a *(Σ/Q)-transducer*, namely a finite-state machine that given inputs in Σ, generates outputs in Q. Formally, the HD strategy is given by $g = \langle \Sigma, Q, M, m_{init}, \rho, \tau\rangle$, where M is a finite set of states, $m_{init} \in M$ is an initial state, $\rho : M \times \Sigma \to M$ is a transition function that is guarded by the inputs in Σ, and $\tau : M \to Q$ labels each state by a state in Q. We extend ρ to finite words by $\rho(m, \epsilon) = m$ and $\rho(m, u \cdot a) = \rho(\rho(m, u), a)$, for all $u \in \Sigma^*$ and $a \in \Sigma$. Then, for $u \in \Sigma^*$, we let $g(u) \in Q$ be the state suggested by g on u. I.e., $g(u) = \tau(\rho(m_{init}, u))$. We

[3] Note also that there can be green transitions among states that belong to the same black component, for example the transition $\langle q_1, a, q_1\rangle$ in Example 1 is a green transition inside $\mathit{black}(q_1)$.

refer to the states of g as *memories*, and say that $m \in M$ is a memory of a state $q \in Q$ if $\tau(m) = q$. We assume that all memories in HD strategies are reachable. Recall that g is an HD strategy for $\mathcal{A}$, and so in particular induce legal runs of $\mathcal{A}$. I.e., for all $m, m' \in M$ and $\sigma \in \Sigma$ such that $m' = \rho(m, \sigma)$, we have that $\langle \tau(m), \sigma, \tau(m') \rangle$ is a transition of $\mathcal{A}$.

Recall that the labeling function $\tau : M \to Q$ of g maps each memory in g to its state in $\mathcal{A}$. We extend τ to sets of memories in the expected way. Thus, for $C \subseteq M$, we have that $\tau(C) = \{\tau(m) : m \in C\}$.

Let $\mathcal{A}_g$ be the tDBW obtained by letting $\mathcal{A}$ follow g. Formally, $\mathcal{A}_g = \langle \Sigma, M, m_{init}, \rho, \alpha_g \rangle$, where $\alpha_g = \{\langle m, \sigma, m' \rangle \in M \times \Sigma \times M : m' = \rho(m, \sigma)$ and $\langle \tau(m), \sigma, \tau(m') \rangle \in \alpha\}$. The following observations follow easily from the definitions and the fact we assume that all memories are reachable.

Lemma 1. *[4] Consider an HD-tNBW $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$ and let $g = \langle \Sigma, Q, M, m_{init}, \rho, \tau \rangle$ be a strategy witnessing its HDness.*

1. *For every memory $m \in M$ in $\mathcal{A}_g$, we have that $L(\mathcal{A}_g^m) = L(\mathcal{A}^{\tau(m)})$.*
2. *For every two memories $m, m' \in M$ in $\mathcal{A}_g$ with $\tau(m) = \tau(m')$, we have that $L(\mathcal{A}_g^m) = L(\mathcal{A}_g^{m'})$.*

The notion of *tightness* for HD automata and strategies is introduced in [4] for Rabin automata with state-based acceptance. Here, we focus on the special case of Büchi automata and a setting with transition-based acceptance. An HD-tNBW $\mathcal{A}$ is *tight* with respect to an HD strategy g if all the transitions of $\mathcal{A}$ are used by g, and for all memories m and m' with $\tau(m) = \tau(m')$, if $m \neq m'$, then there is a black path from m to m' in $\mathcal{A}_g$. Equivalently, if all memories of a state $q \in Q$ belong to the same black component of $\mathcal{A}_g$. We say that $\mathcal{A}$ is *tight* if it is tight with respect to some HD strategy.

Tightness is achieved by starting with some witness strategy g, and then repeatedly (1) removing from $\mathcal{A}$ transitions that are not used by g and (2) merging memories m and m' of the same state such that there is no black path from m' to m. Intuitively, by removing m and redirecting transitions that go to m into m', we save one memory of $\tau(m)$ and we continue to accept all words in $L(\mathcal{A})$, as between successive traversals of transitions that have been redirected, there is at least one traversal of a transition between different black components. As $\mathcal{A}$ is normal, the latter are green. Formally, we have the following.

Lemma 2. *[4] Every HD-tNBW $\mathcal{A}$ can be made tight.*

Proof. Let $g = \langle \Sigma, Q, M, m_{init}, \rho, \tau \rangle$ be a strategy witnessing $\mathcal{A}$'s HDness. To make $\mathcal{A}$ tight with respect to g, we repeatedly apply the following modifications to $\mathcal{A}$ and g:

1. Remove all transitions of $\mathcal{A}$ that are not used by g.
2. If there are memories $m \neq m'$ with $\tau(m) = \tau(m')$ such that m is unreachable from m' in $\mathcal{A}_g$ or m is reachable from m' only via paths that visit α_g, then modify g by removing the memory m and redirecting transitions to m to m' (in particular, if m is the initial memory of g, then we make m' initial).

Before we prove that the modifications result in an equivalent and tight HD-tNBW, we note that they only involve removal of transitions in $\mathcal{A}$. Indeed, the first step clearly only removes transitions, and the second only redirects transitions in the strategy g. In more details, the redirection in the second step is of transitions of the form $\langle m'', \sigma, m\rangle$ to $\langle m'', \sigma, m'\rangle$ for memories m and m' with $\tau(m) = \tau(m') = s$. Accordingly, both transitions follow the transition $\langle \tau(m''), \sigma, s\rangle$ in $\mathcal{A}$, and so the redirection in g follows transitions in $\mathcal{A}$.

We prove that in each iteration, if we start with an HD-NBW $\mathcal{A}$ and a strategy g that witnesses its HDness, the modification results in an HD-tNBW $\mathcal{A}'$ and function g' such that $L(\mathcal{A}') = L(\mathcal{A})$ and g' witnesses the HDness of $\mathcal{A}'$. Then, since the modification of $\mathcal{A}$ into $\mathcal{A}'$ at most removes transitions from $\mathcal{A}$, it follows that $\mathcal{A}'$ is an equivalent HD-tNBW that is embodied in $\mathcal{A}'$.

It is easy to see that the above holds for modifications in the first step. Indeed, since removal of transitions that are not used by g does not restrict the language of $\mathcal{A}$, we have that $L(\mathcal{A}') = L(\mathcal{A})$, and g stays an HD strategy for $\mathcal{A}'$.

We proceed to the modifications in the second step, thus the replacement of g by g'. There, we prove that $L(\mathcal{A}'_{g'}) \subseteq L(\mathcal{A}_g)$, implying that $L(\mathcal{A}') = L(\mathcal{A})$ with g' being an HD strategy for $\mathcal{A}'$.

Let m and m' be memories in $\mathcal{A}_g$ such that g' is obtained from g by removing the memory m and redirecting to m' the transitions to m. Consider an accepting run r_g of $\mathcal{A}_g$ of g on some word $w \in L(\mathcal{A}_g)$. Let $r_{g'} = \langle m'_0, \sigma_1, m'_1\rangle, \langle m'_1, \sigma_2, m'_2\rangle, \ldots$ be the run of $\mathcal{A}_{g'}$ on w. We prove that $r_{g'}$ is accepting.

Let $k \geq 0$ be the number of times that $r_{g'}$ takes redirected transitions to m'. Formally, $k = |\{i \geq 0 : \rho(m'_i, \sigma_{i+1}) = m\}|$. Note that if m is not reachable from m', then once $r_{g'}$ visits m', it never reaches m, and so it never takes a redirected transition. Hence, if $k = \infty$, the reason for the modification in the second step is that m is reachable from m' only via green runs. Therefore, $r_{g'}$ traverses at least one green transition between each pair of successive redirected transitions. It follows that when $k = \infty$, we have that $r_{g'}$ is accepting.

We continue to the case $1 \leq k < \infty$. Let $i \geq 0$ be the last index in $r_{g'}$ in which a redirected transition is taken. Thus, $i \geq 0$ is such that $m'_i = m'$ and for all $j > i$, the j-th transition in $r_{g'}$ is an original transition of g. That is, $\rho(m'_{j-1}, \sigma_j) = m'_j$, for all $j > i$. Accordingly, the suffix of $r_{g'}$ from the i-th position coincides with the run of $\mathcal{A}_g^{m'}$ on the suffix $w[i+1, \infty]$. This, it is left to show that $\mathcal{A}_g^{m'}$ accepts $w[i+1, \infty]$. The strategy g' witnesses that $\tau(m') \in \delta(q_0, w[1, i])$. Since $w \in L(\mathcal{A})$, it follows by the SDness of $\mathcal{A}$ that $w[i+1, \infty] \in L(\mathcal{A}^{\tau(m')})$. Then, since all memories are reachable, Lemma 1 implies that $w[i+1, \infty] \in L(\mathcal{A}_g^{m'})$. Therefore, $\mathcal{A}_g^{m'}$ accepts $w[i+1, \infty]$, and we are done. □

Tight HD-tNBWs have some nice properties. For example, tight HD-tNBWs are SD. To see why, observe that if $\mathcal{A}$ has two transitions $\langle q, \sigma, s_1\rangle, \langle q, \sigma, s_2\rangle \in \Delta$ with $s_1 \not\sim s_2$, then no strategy that witnesses the HDness of $\mathcal{A}$ can use both transitions [12]. Indeed, if $\langle q, \sigma, s_1\rangle$ is used by an HD strategy, then $L(s_2) \subseteq L(s_1)$.

Another advantage of a tight strategy g is that the partition into black components in $\mathcal{A}$ induces a partition to black components in $\mathcal{A}_g$. Formally, we have the following.

Lemma 3. *If $\mathcal{A}$ is tight with respect to g, then for all memories $m, m' \in M$, it holds that $\tau(m') \in \mathit{black}(\tau(m))$ iff $m' \in \mathit{black}(m)$.*

Proof. Consider an HD-tNBW $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$ that is tight with respect to $g = \langle \Sigma, Q, M, m_{init}, \rho, \tau \rangle$. We prove that for all memories $m, m' \in M$, it holds that $\tau(m')$ is reachable via a black run of $\mathcal{A}^{\tau(m)}$ iff m' is reachable via a black run of $\mathcal{A}_g^m$. By the symmetry between m and m' and the definition of black components, it follows that $\tau(m') \in \mathit{black}(\tau(m))$ iff $m' \in \mathit{black}(m)$.

Assume first that m' is reachable via a black run of $\mathcal{A}_g^m$. Consider a black run $r = \langle m_0, \sigma_1, m_1 \rangle, \langle m_2, \sigma, m_3 \rangle, \ldots, \langle m_{k-1}, \sigma_k, m_k \rangle$ of $\mathcal{A}_g^m$ to m', thus $m_0 = m$ and $m_k = m'$. By definition of α_g, we have that $\langle \tau(m_i), \sigma_{i+1}, \tau(m_{i+1}) \rangle \notin \alpha$ for all $0 \leq i \leq k-1$. Hence, the corresponding run mapped by τ is a black run of $\mathcal{A}^{\tau(m)}$ to $\tau(m')$, as required.

Assume now that $\tau(m')$ is reachable via a black run of $\mathcal{A}^{\tau(m)}$. If $\tau(m) = \tau(m')$, then m and m' are both memories of the same state and since g is tight it follows that $m' \in \mathit{black}(m)$. In particular, m' is reachable via a black run in $\mathcal{A}_g^m$ as required. Otherwise, the black run from $\tau(m)$ to $\tau(m')$ is a non empty run. That is, there exists a black run $r = \langle p_0, \sigma_1, p_1 \rangle, \langle p_2, \sigma, p_3 \rangle, \ldots, \langle p_{k-1}, \sigma_k, p_k \rangle$ of $\mathcal{A}^{\tau(m)}$ to $\tau(m')$, thus $p_0 = \tau(m)$ and $p_k = \tau(m')$. Since g is tight, all transitions are used by g, and hence there are memories $m'_0, m_1, m'_1, m_2, m'_2, \ldots m'_{k-1}, m_k \in M$, such that $\langle m'_{i-1}, \sigma_i, m_i \rangle$ is a black transition of $\mathcal{A}_g$ and $\langle \tau(m'_{i-1}), \sigma_i, \tau(m_i) \rangle = \langle p_{i-1}, \sigma_i, p_i \rangle$ for all $1 \leq i \leq k$. Let $m_0 = m$ and $m'_k = m'$. Since for all $0 \leq i \leq k$ the memories m_i and m'_i are both memories of the same state, it follows by tightness that there are words $w_0, w_1, w_2, \ldots, w_k \in \Sigma^*$ such that the run of $\mathcal{A}_g^{m_i}$ over w_i is black and reaches m'_i. It then follows that the run of $\mathcal{A}_g^{m_0}$ on the word $w_0 \cdot \sigma_1 \cdot w_1 \cdot \sigma_2 \cdots \sigma_k \cdot w_k$ is black and reaches m'_k. That is, $m' = m'_k$ is reachable via a black run in $\mathcal{A}_g^m = \mathcal{A}_g^{m_0}$ and we are done. □

An immediate consequence of Lemma 3 is that if $\mathcal{A}$ is normal and tight with respect to g, then $\mathcal{A}_g$ is normal as well. Indeed, if $\langle m, \sigma, m' \rangle$ is black, then $\langle \tau(m), \sigma, \tau(m') \rangle$ is black, and so since $\mathcal{A}$ is normal we have that $\tau(m') \in \mathit{black}(\tau(m))$. Then, by Lemma 3 it follows that $m' \in \mathit{black}(m)$.

3.2 Fast memories

Consider a normal HD-tNBW $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$ that is tight with respect to a strategy $g = \langle \Sigma, Q, M, m_{init}, \rho, \tau \rangle$. Let $\mathcal{A}_g = \langle \Sigma, M, m_{init}, \rho, \alpha_g \rangle$.

For a memory $m \in M$ and a state $q \in Q$, we say that m *covers* q if for every word $w \in \Sigma^+$, and run r from q on w, if r is green, then so is the run of $\mathcal{A}_g$ from m on w. Thus, m covers q if for every word $w \in \Sigma^+$, runs of $\mathcal{A}^q$ on w cannot traverse an α transition before the run of $\mathcal{A}_g^m$ on w traverses an α_g transition. Intuitively, if m covers q, and $\tau(m)$ is a state equivalent to q, then following the

strategy g from m is better than following any other run from $\mathcal{A}^q$, in the sense that the former traverses an α transition faster.

It is not difficult to see that the cover relation is transitive in the sense that if m covers $\tau(m')$ and m' covers q, then m covers q. Formally we have the following.

Lemma 4. *Consider states $q, p \in Q$ and memories $m, m_q \in M$, such that m_q is a memory of q. If m covers q and m_q covers p, then m covers p.*

Proof. Consider a word $w \in \Sigma^+$ for which there exists a green run of $\mathcal{A}$ from p. We need to show that the run of $\mathcal{A}_g$ on w from m is green as well. Since m_q covers p it follows that the run of $\mathcal{A}_g$ on w from m_q is green. So m_q witnesses that there exists a green run of $\mathcal{A}$ on w from q. Then similarly, as m covers q it follows that the run of $\mathcal{A}_g$ on w from m is green and we are done. □

In contrast to the transitivity in Lemma 4, the covering relation needs not be reflexive. Thus, there may be a memory m, a word $w \in \Sigma^+$, and a run of $\mathcal{A}^{\tau(m)}$ on w that traverses a green transition while the run of $\mathcal{A}^m$ on w is black. We say that a memory m is *fast* if m covers $\tau(m)$. We then say that a state $q \in Q$ is *fast* if there exists a fast memory of q. It is not hard to see that the property of being fast is preserved along black paths. Formally, we have the following.

Lemma 5. *For all states $q \in Q$, if q is fast, then all states in $black(q)$ are fast.*

Proof. Consider a fast memory m_q of q. We prove that all the memories in $black(m_q)$ are fast. Since, by Lemma 3, each state $q' \in black(q)$ has a memory $m_{q'} \in black(m_q)$ with $\tau(m_{q'}) = q'$, the claim follows.

Consider a memory $m \in black(m_q)$. Let $m_0, \ldots, m_l$ be a path from m_q to m in $black(m_q)$, thus $m_0 = m_q$ and $m_l = m$. We prove that m is fast. The proof proceeds by an induction on l. First, if $l = 0$, then $m = m_q$, and we are done. For $l > 0$, consider the black transition $\langle m_{l-1}, \sigma, m_l \rangle$. Let $p_{l-1} = \tau(m_{l-1})$ and $p_l = \tau(m_l)$. By the induction hypothesis, the memory m_{l-1} is fast, and so m_{l-1} covers p_{l-1}. Hence, as $\langle m_{l-1}, \sigma, m_l \rangle$ is black, the transition $\langle p_{l-1}, \sigma, p_l \rangle$ cannot be green, and so it is black.

Assume towards contradiction that m_l is not fast. That is, m_l does not cover p_l. Then, there exists a word $w \in \Sigma^+$ on which there exists a green run r_l from p_l, while the run of $\mathcal{A}_g^{m_l}$ on w is black. Consider the word $\sigma \cdot w$. The sequence $r = \langle p_{l-1}, \sigma, p_l \rangle \cdot r_l$ is a green run of $\mathcal{A}^{p_{l-1}}$ on $\sigma \cdot w$. On the other hand, since $\langle m_{l-1}, \sigma, m_l \rangle$ is black, the run of $\mathcal{A}_g^{m_{l-1}}$ on $\sigma \cdot w$ is black. Thus, the word $\sigma \cdot w$ witnesses that m_{l-1} does not cover p_{l-1}, and we have reached a contradiction to m_{l-1} being fast. □

We now argue that each memory covers an equivalent fast state. We first need the following lemma.

Lemma 6. *Consider a state $q \in Q$. For every memory m_q of q, there is a state s equivalent to q such that m_q covers s.*

Proof. Let $\mathcal{A}_g = \langle \Sigma, M, m_{init}, \rho, \alpha_g \rangle$. Consider a memory m_q of q. Assume towards contradiction that for all states s equivalent to q it holds that m_q does not cover s. We prove that for all $s_0 \sim_{\mathcal{A}} q$ we can define a word $x_1 \in \Sigma^+$ and a state $s_1 \in Q$ equivalent to q, such that the following holds:

1. The run of $\mathcal{A}_g$ on x_1 from m_q is a black cycle. I.e., the run of $\mathcal{A}_g$ on x_1 from m_q does not traverse α_g transitions and is such that $\rho(m_q, x_1) = m_q$.
2. There is a green run r_1 on x_1 from s_0 to some state s_1 equivalent to q.

Note that then, as $q \sim_{\mathcal{A}} s_1$, and by assumption m_q does not cover s_1, we can use the same argument to also define $x_2 \in \Sigma^+$, and $s_2 \in Q$ with similar properties. I.e., the run of $\mathcal{A}_g$ on x_2 from m_q is a black cycle, whereas there is a green run r_2 on x_2 from s_1 that reaches a state s_2 equivalent to q. Repeating this argument inductively, we end up with an infinite word $w = x_1 \cdot x_2 \cdot x_3 \cdots$ such that, on the one hand, the run of $\mathcal{A}_g$ from m_q on w does not traverse α_g, and so $w \notin L(\mathcal{A}_g^{m_q}) = L(\mathcal{A}^q)$, and on the other hand, the concatenation of the runs $r_1, r_2, \ldots$ on the subwords $x_1, x_2, \ldots$, is a run r of $\mathcal{A}^{s_0}$ on w that traverses infinitely many green transitions. Recall that a run is rejecting iff it is eventually black, and so the induced run r is accepting, and we reached a contradicting to the fact that $q \sim_{\mathcal{A}} s_0$. Taking $s_0 = q$ completes the proof.

We define x_1 as follows. As m_q does not cover s_0, there is a word $y_1 \in \Sigma^+$ such that the run of $\mathcal{A}_g$ from m_q on y_1 is black, yet there is a green run of $\mathcal{A}$ from s_0 on y_1 that reaches some state s_1'. Let $m_1 = \rho(m_q, y_1)$. Recall that $\mathcal{A}_g$ is normal, and so $m_1 \in \mathit{black}(m_q)$. Hence, there is a word $z_1 \in \Sigma^*$ such that the run of $\mathcal{A}_g$ from m_1 on z_1 is black and reaches m_q. We define $x_1 = y_1 \cdot z_1$, and $s_1 \in Q$ to be some state reachable from s_1' with a run on z_1. By the above, the run of $\mathcal{A}_g$ on x_1 from m_q is a black cycle, and there is a green run of $\mathcal{A}$ on x_1 from s_0 to s_1. As $q \sim s_0$, and $q \in \delta(q, x_1)$ and $s_1 \in \delta(s_0, x_1)$, the SDness property implies that s_1 is equivalent to q. □

Since M is finite, the transitivity of the "covers" relation (Lemma 4), together with Lemma 6, imply the following.

Lemma 7. *For every memory m, there is a fast state p_m such that p_m is equivalent to $\tau(m)$ and m covers p_m. In particular, every state in Q has an equivalent fast state in Q.*

Intuitively, Lemma 7 implies that the set of fast states of $\mathcal{A}$ can represent the languages of all the states of $\mathcal{A}$. In Lemma 8 below we show that this representation can be carried on to a small set of fast memories – one for each fast state.

Lemma 8. *Consider an HD strategy g. For every state $q \in Q$, if q is fast, then we can modify g so that it has only a single memory of q.*

Proof. Let $g = \langle \Sigma, Q, M, m_{init}, \rho, \tau \rangle$. Consider a fast state $q \in Q$, and let m_q be a fast memory of q. We define a strategy g' in which m_q is the only memory of q. Essentially, all the transitions of g that lead to other memories of

q are directed in g' to m_q. Formally, $g' = \langle \Sigma, Q, M', m'_{init}, \rho', \tau \rangle$, where M' is obtained by removing from M all the memories of q but m_q, the initial state m'_{init} stays m_{init}, unless it is a memory of q that is no longer in M', in which case $m'_{init} = m_q$, and ρ' is such that for all $m \in M$ and $\sigma \in \Sigma$, if $\tau(\rho(m, \sigma)) = q$, then $\rho'(m, \sigma) = m_q$, and otherwise $\rho'(m, \sigma) = \rho(m, \sigma)$.

We prove that $L(\mathcal{A}_g) = L(\mathcal{A}_{g'})$. For that, we first prove that m_q is fast also with respect to g'. That is, for all words $w \in \Sigma^+$, if there is a green run of $\mathcal{A}^q$ on w, then the run of $\mathcal{A}_{g'}^{m_q}$ on w is green. Assume by way of contradiction that m_q is not fast with respect to g' and let $w = \sigma_1 \cdot \sigma_2 \cdots \sigma_k \in \Sigma^+$ be a minimal witness for that. That is, there is a green run of $\mathcal{A}^q$ on w, the run r' of $\mathcal{A}_{g'}^{m_q}$ on w is black, and there is no word shorter than w for which the above hold.

As m_q is fast with respect to g, the run $r = \langle r_0, \sigma_1, r_1 \rangle, \ldots, \langle r_{k-1}, \sigma_k, r_k \rangle$ of $\mathcal{A}_g^{m_q}$ on w is green. Consider the black run $r' = \langle r'_0, \sigma_1, r'_1 \rangle, \ldots, \langle r'_{k-1}, \sigma_k, r'_k \rangle$ of $\mathcal{A}_{g'}^{m_q}$ on w, and consider the minimal index $1 \leq i \leq k$ in which r and r' differ. Thus, for all $1 \leq j < i$, we have that $r_j = r'_j$, $r'_i = m_q$, and $r_i = m'_q$ for some memory m'_q of q with $m'_q \neq m_q$. Note that such an index i must exist. In fact, $i < k$, as otherwise the runs of $\mathcal{A}^q$ on w generated by r and r' are identical, contradicting the fact that the former run is green whereas the latter is black.

We claim that the suffix $w' = \sigma_{i+1} \cdot \sigma_{i+2} \cdots \sigma_k$ of w also witnesses that m_q is not fast with respect to g', contradicting the minimality of w. Since i is minimal and the run r' is black, the fact that the run r is green implies that r traverses a green transition only after its first i transitions, thus when it reads w' from m'_q. Hence, there is a green run of $\mathcal{A}^q$ on w'. On the other hand, the run of $\mathcal{A}_{g'}^{m_q}$ on w' is a suffix of the run r', and is black. If follows that w' is a shorter witness to m_q not being fast, and we have reached a contradiction.

We can now prove that $L(\mathcal{A}_{g'}) = L(\mathcal{A}_g)$. Since $L(\mathcal{A}_{g'}) \subseteq L(\mathcal{A}) = L(\mathcal{A}_g)$, it is enough to prove that $L(\mathcal{A}_g) \subseteq L(\mathcal{A}_{g'})$. Consider a word $w \in L(\mathcal{A}_g)$, and let $r = \langle m_0, \sigma_1, r_m \rangle, \langle m_1, \sigma_2, m_2 \rangle, \ldots$ be the accepting run of $\mathcal{A}_g$ on w. We show that the run $r' = \langle m'_0, \sigma_1, m'_1 \rangle, \langle m'_1, \sigma_2, m'_2 \rangle, \ldots$ of $\mathcal{A}_{g'}$ on w is accepting to. To begin with, as memories of the same state are equivalent in $\mathcal{A}_g$, the SDness property implies that for all $i \geq 0$, we have that $L(\mathcal{A}_g^{m_i}) = L(\mathcal{A}_g^{m'_i})$. We say that an index $i \geq 0$ is a *jumping index* if r' uses the memory m_q at m'_i instead of some other memory of q. Thus, $m'_i = m_q$ whereas $m_i \neq m_q$.

For $i \geq 0$ let $r[i, \infty]$ be the suffix of r from the i-th tranistion. Thus, $r[i, \infty]$ is the run of $\mathcal{A}_g^{m_i}$ on $w[i+1, \infty]$. The run $r'[i, \infty]$ is defined similarly with respect to r'. Assume first that r' has finitely many jumping indices, and let $j \geq 0$ be the last such index. Then, the run $r'[j, \infty]$ coincides with the run of $\mathcal{A}_g^{m'_j}$ on the suffix $w[j+1, \infty]$. By the SDness property, we have that $\mathcal{A}_g^{m_j} \sim \mathcal{A}_g^{m'_j}$. Since $r[j, \infty]$ is an accepting run of $\mathcal{A}_g^{m_j}$ on $w[j+1, \infty]$, we conclude that $w[j+1, \infty] \in \mathcal{A}_g^{m'_j}$, and thus r' is an accepting run on w.

Now, if the run r' has infinitely many jumping indices, we claim that r' traverses infinitely many green transitions, and is thus accepting. Assume towards contradiction that r' traverses only finitely many green transitions. Thus, there is a jumping index $i \geq 0$ such that $r'[i, \infty]$ is a black run on the suffix $w[i+1, \infty]$.

As $L(\mathcal{A}_g^{m_i}) = L(\mathcal{A}_g^{m'_i})$ and $w[i+1,\infty] \in L(\mathcal{A}_g^{m_i})$, there is a green run of $\mathcal{A}_g^{m'_i}$ on $w[i+1,j]$, for some $j \geq i+1$. Then, since i is a jumping index, we have that $m'_i = m_q$. Since m_q is fast with respect to g', then $r'[i,\infty]$ traverses a green transition upon reading $w[i+1,j]$, contradicting the fact that $r'[i,\infty]$ is black . □

4 An Efficient Complementation Construction

In this section, we present our efficient complementation construction. As in [12], our starting point is an HD-tNBW[4] $\mathcal{A}$ and a finite-state HD strategy for $\mathcal{A}$. In [12], the strategy is manipulated until every state of $\mathcal{A}$ has only linearly many "copies" in the obtained HD strategy, implying that the strategy can be viewed as an equivalent deterministic automaton of quadratic size. Here, the "copies" are induced by the fast states $\mathcal{A}$, which, as we argue below, can be identified efficiently. Also, while [12] uses the manipulated strategy only for determinization, we observe that it induces also an HD-tNCW of linear size that complemnts $\mathcal{A}$, and so we gain both a linear complementation and a quadratic determinization constructions.

Consider an HD-tNBW $\mathcal{A}$. We construct an HD-tNCW that complements $\mathcal{A}$ in two steps: (1) Trimming redundant transitions from $\mathcal{A}$, and (2) Defining a co-Büchi acceptance condition on top of the state space of the trimmed HD-tNBW.

The correctness of the construction follows from the analysis in Section 3. Essentially, we show that trimming of redundant transitions makes the identification of fast memories easy, and that the complementing HD-tNCW can be defined on top of the state space of fast states.

Consider an HD-tNBW $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$. We say that $\mathcal{A}$ is *maximally-trimmed* if we cannot remove transitions from $\mathcal{A}$ and obtain an equivalent HD-tNBW. Note that the latter condition means that for every $t \in \Delta$, the automaton obtained from $\mathcal{A}$ by removing t from Δ either rejects some words that $\mathcal{A}$ accepts, or has the same language as $\mathcal{A}$, but is no longer HD. Intuitively, transitions in maximally-trimmed HD-tNBWs are used by every HD strategy.

Theorem 1. *Every HD-tNBW can be made maximally-trimmed in polynomial time.*

Proof. Consider an HD-tNBW $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$. We repeatedly remove transitions from Δ while maintaining $\mathcal{A}$'s language and HDness. Formally, let $t_1, \ldots, t_m$ be an order on the transitions in Δ. Starting with $i = 1$, we check whether the tNBW obtained from $\mathcal{A}$ by removing t_i is an equivalent HD-tNBW. If so, we remove t_i. In any case, we increase i by 1. Since removal of a transition cannot make another transition redundant, the order in which the transitions are

[4] The construction in [12] starts with an HD-NBW, thus one with *state-based* acceptance. Since every HD-NBW can be translated to an equivalent HD-tNBW on the same structure, our setting is more general.

checked is not important. [5] Hence, by definition, once $i = m + 1$, the procedure terminates and we end up with a maximally-trimmed HD-tNBW.

Since checking HDness and language equivalence for HD-tNBWs can be done in polynomial time [2], and since the number of times each transition in Δ is examined is linear in $|\Delta|$, the whole procedure is polynomial in $\mathcal{A}$. Note that normalizing a maximally-trimmed HD-NBW leaves it maximally-trimmed. Indeed, normalization does not change the classification of runs to accepting or rejecting, and so a transition t can be removed after normalization iff there is an HD strategy that does not use t also before normalization. □

Consider an HD-tNBW $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$. We say that a state $q \in Q$ is *deterministic* if for all letters $\sigma \in \Sigma$, it holds that $|\delta(q, \sigma)| = 1$. A set $B \in \mathcal{B}(\mathcal{A})$ is a *black deterministic component* if all states $q \in B$ are deterministic.

Lemma 9. *Consider a maximally trimmed HD-tNBW $\mathcal{A}$. For every state q of $\mathcal{A}$, we have that q is fast iff $black(q)$ is a deterministic black component.*

Proof. First, as detailed in the proof of Lemma 2, making an HD-tNBW tight involves removal of transitions as long as some condition, which is stronger than the one used in maximal trimming, is valid. Hence, all maximally trimmed HD-tNBW are tight, and so the analysis in Section 3 applies to them. Let g be the HD-strategy of $\mathcal{A}$.

We first prove that if $black(q)$ is deterministic, then q is fast. Let m be a memory of q. Assume that a word $w \in \Sigma^+$ is such that the run of $\mathcal{A}_g^m$ on w is black. Then, as transitions in $\mathcal{A}_g$ are induced by transitions in $\mathcal{A}$, there exists a black run of $\mathcal{A}^q$ on w. Since $black(q)$ is deterministic, there is a unique run of $\mathcal{A}^q$ on w, and so all runs of $\mathcal{A}^q$ are black. It follows that if $\mathcal{A}^q$ has a green run on $w \in \Sigma^+$, then the run of $\mathcal{A}_g^m$ on w is green. That is, m covers q and since $\tau(m) = q$ we have that q is fast.

We continue to the second direction and prove that if q is fast, then $black(q)$ is deterministic. Consider a fast state q, and assume by way of contradiction that $black(q)$ is not deterministic. There is a state p in $black(q)$ and a letter $\sigma \in \Sigma$ such that $\delta(p, \sigma)$ is not a singleton. By Lemma 5, the state p is fast too. Also, by Lemma 8, we can obtain from g an HD strategy g' in which p has a unique memory m. We claim that the strategy g' witnesses that a maximal-trimming of $\mathcal{A}$ should have made p deterministic. Formally, consider the HD-tNBW $\mathcal{A}'$ obtained from $\mathcal{A}$ by trimming from $\mathcal{A}$ all the σ-successors of p but the one used by m. The strategy g' witnesses that $\mathcal{A}'$ is an HD-tNBW equivalent to $\mathcal{A}$. Thus, $\mathcal{A}$ is not maximally trimmed, and we reached a contradiction. □

We can now describe the complementation construction.

Theorem 2. *Let $\mathcal{A}$ be a maximally-trimmed HD-tNBW. We can construct in polynomial time an HD-tNCW $\mathcal{C}$ such that $L(\mathcal{C}) = \overline{L(\mathcal{A})}$, and the state space of $\mathcal{C}$ is contained in the state space of $\mathcal{A}$.*

[5] Note that different orders may lead to different HD-tNBWs, but they are all maximally trimmed.

Proof. Let $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$. Recall that $\mathcal{A}$ is normal, implying that black transitions from black components stay in the same black component. We define the HD-tNCW $\mathcal{C} = \langle \Sigma, Q', q_0', \delta', \beta \rangle$ as follows.

- $Q' = \{q \in Q : q$ belongs to a deterministic black component of $\mathcal{A}\}$.
- The state $q_0' \in Q'$ is chosen such that $q_0 \sim_{\mathcal{A}} q_0'$.
- Since Q' contains only deterministic states of $\mathcal{A}$, we know that for all $q \in Q'$ and $\sigma \in \Sigma$, we have that $\delta(q, \sigma) = \{s\}$, for some state $s \in Q$. The transition function $\delta'(q, \sigma)$ then depends on whether s belongs to Q', and is defined as follows.
  - (original transitions) If $s \in Q'$, then $\delta'(q, \sigma) = \{s\}$. In this case, we have that $\langle q, \sigma, s \rangle \in \beta$ iff $\langle q, \sigma, s \rangle \in \alpha$.
  - (new transitions) If $s \notin Q'$, then $\delta'(q, \sigma) = \{p \in Q' : p \sim_{\mathcal{A}} s\}$. In this case, $\langle q, \sigma, p \rangle \in \beta$ for all $p \in \delta'(q, \sigma)$.

  Thus, if s is in Q', then it is the only σ-successor of q, and the transition to it is in β iff the corresponding transition of $\mathcal{A}$ is in α. If s is not in Q', then the σ-successors of q in $\mathcal{C}$ are all the states in Q' that are $\sim_{\mathcal{A}}$-equivalent to s, and the transition to them is in β.

Since checking language equivalence for HD-tNBWs can be done in polynomial time, so are the checks required for the definition of the new transitions.

We first prove that $\mathcal{C}$ is well defined. By Lemma 9, the set Q' is the set of fast states/ By Lemma 7, for every $q \in Q$ there exists a states $s \in Q'$ equivalent to q. Hence, the initial state q_0' is well defined, and $\delta'(q, \sigma)$ is non-empty for all $q \in Q'$ and $\sigma \in \Sigma$. Moreover, a transition $\langle p, \sigma, p' \rangle$ of $\mathcal{C}$ belongs to β iff the unique σ transition from p in $\mathcal{A}$ is green and hence belongs to α.

We prove that $\mathcal{C}$ is an HD-tNCW with $L(\mathcal{C}) = \overline{L(\mathcal{A})}$. Note that by the SDness of $\mathcal{A}$, the definition of δ', and the choice of q_0' to be one that is equivalent to q_0, we have that that for all $u \in \Sigma^*$, all states in $\delta'(q_0', u)$ are $\sim_{\mathcal{A}}$-equivalent to all states in $\delta(q_0, u)$ (the proof is by an easy induction on $|u|$). In particular, the state $g(u) \in \delta(q_0, u)$, reached by g on u, is equivalent to all states in $\delta(q_0, u) \cup \delta'(q_0', u)$. So overall, for all $q \in \delta(q_0, u)$ and for all $p \in \delta'(q_0', u)$ it holds that $q \sim_{\mathcal{A}} g(u) \sim_{\mathcal{A}} p$. We refer to this as the "equivalence property".

We first prove that $L(\mathcal{C}) \subseteq \overline{L(\mathcal{A})}$. Let $r = \langle q_0', \sigma, q_1' \rangle, \langle q_1', \sigma_2, q_2' \rangle, \ldots$ be an accepting run of $\mathcal{C}$ on some word $w = \sigma_1 \cdot \sigma_2 \cdot \sigma_3 \cdots \in \Sigma^\omega$. Let $i \geq 0$ be such that r does not traverse β after time i. I.e., $\langle q_j', \sigma_{j+1}, q_{j+1}' \rangle \notin \beta$ for all $j \geq i$. By definition of β this means that all transitions after time i are original black transitions of $\mathcal{A}$, and so the suffix of r from the i-th position is a rejecting run of $\mathcal{A}^{q_i'}$ on the suffix $w[i+1, \infty]$. Let $q = g(w[1, i])$. Then, by the equivalence property, we have $q_i' \sim_{\mathcal{A}} q$ and so $w[i+1, \infty] \notin L(\mathcal{A}^q) = L(\mathcal{A}^{q_i'})$. Thus, by the SDness of $\mathcal{A}$ the latter implies that $w \notin L(\mathcal{A})$ and we are done with the first direction.

We continue and prove that $\mathcal{C}$ is HD and $\overline{L(\mathcal{A}_g)} \subseteq L(\mathcal{C})$. Since $L(\mathcal{A}_g) = L(\mathcal{A})$ this will complete the second direction. We do so by defining a strategy $f : \Sigma^* \to Q'$ such that for all $w \notin L(\mathcal{A}_g)$, we have that $f(w)$ is an accepting run of $\mathcal{C}$ on w. We define f to be such that we always take original transitions of $\mathcal{A}$ as long as possible, and whenever all σ-transitions from $f(u)$ are new transitions, we pick

one that is to a fast state covered by $g(u \cdot \sigma)$. Such a state exists by Lemma 7. Formally, for a memory $m \in M$, let p_m be some fast state equivalent to $\tau(m)$ such that m covers p_m. Let $f(\epsilon) = q'_0$, and for all $u \in \Sigma^*$ and $\sigma \in \Sigma$, let $s \in Q$ and $m \in M$ be such that $\delta(f(u), \sigma) = \{s\}$ and $m = \rho(m_{init}, u \cdot \sigma)$, and define $f(u \cdot \sigma)$ as follows:

$$f(u \cdot \sigma) = \begin{cases} s & \text{if } s \in Q' \\ p_m & \text{otherwise} \end{cases}$$

Note that in the case that $s \notin Q'$ then $\delta'(q, \sigma) = \{p \in Q' : p \sim_{\mathcal{A}} s\}$ and $p_m \sim_{\mathcal{A}} \tau(m) = g(u \cdot \sigma)$, and by the equivalence property $g(u \cdot \sigma) \sim_{\mathcal{A}} p$ for all $p \in \delta'(q, \sigma)$. Thus, the fast state p_m indeed belongs to $\delta'(q, \sigma)$, and f is well defined.

Consider a word $w \notin L(\mathcal{A}_g)$ and let r_g be the run of $\mathcal{A}_g$ on w. For $j \geq 0$, let $m_j = \rho(m_{init}, w[1, j])$ be the j-th memory reached by r_g on w. Similarly, let r_f be the run of $\mathcal{C}$ generated by f on w, and, for $j \geq 0$, let $q'_j = f(w[1, j])$ be the j-th fast state reached by r_f. By the equivalence property, it holds that $\tau(m_j) \sim_{\mathcal{A}} q'_j$, for all $j \geq 0$. In particular, $w[j+1, \infty] \notin L(\mathcal{A}_g^{m_j}) = L(\mathcal{A}^{q'_j})$.

Moreover, since $w \notin \mathcal{A}_g$, it holds that r_g is eventually black. That is, there exists an index $i_0 \geq 0$, such that after position i_0, the run r_g traverses no more green transitions. Note that if r_f takes finitely many new transitions, then there exists $j \geq 0$ such that the j-th suffix of r_f coincides with the run of $\mathcal{A}^{q'_j}$ on $w[j+1, \infty]$. Since $w[j+1, \infty] \notin L(\mathcal{A}^{q'_i})$, it follows that r_f also takes finitely many original α transitions. Thus, if r_f takes finitely many new transitions, then r_f is co-Büchi accepting with respect to β, which is the union of new transitions and original α transitions. We conclude the proof by proving that r_f takes at most one new transition after position i_0.

Assume that f takes at least one new transition after position i_0. Let $i_1 > i_0$ be the first position it does so. By the definition of f, we have that $m_{i_1} = \rho(m_{init}, w[1, i_1])$ covers q'_{i_1}. Now, as $i_1 > i_0$, the run of $\mathcal{A}_g^{m_{i_1}}$ on $w[i_1 + 1, \infty]$ is black, and since m_{i_1} covers q'_{i_1}, all the runs of $\mathcal{A}^{q'_{i_1}}$ on $w[i_1+1, \infty]$ are black too. In fact, since q'_{i_1} belongs to a deterministic black component, it follows that $\mathcal{A}^{q'_{i_1}}$ has a unique run on $w[i_1 + 1, \infty]$. Since f follows the unique original transitions as long this is possible, the i_1-th suffix of r_f coincides with the run of $\mathcal{A}^{q'_{i_1}}$ on $w[i_1 + 1, \infty]$, and is hence an original run with no new transitions. □

We conclude this section by a determinization construction that makes use of the complementation construction. The construction is based on a construction from [3,14], which generates a deterministic automaton for a language L from HD automata for L and $\overline{L}$. Here, we use a slight variant, where the automata for both L and $\overline{L}$ use transition-based acceptance conditions.

Theorem 3. *Let $\mathcal{A}$ be an HD-tNBW with n states, and $\mathcal{C}$ be an HD-tNCW for $\overline{L(\mathcal{A})}$ with m states. We can determinize $\mathcal{A}$ in polynomial time, obtaining a tDBW with at most $n \cdot m$ states.*

Proof. Let $\mathcal{A} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$, and $\mathcal{C} = \langle \Sigma, Q', q'_0, \delta', \beta \rangle$. Consider the tNBW $\mathcal{P}$ obtained by taking the product of $\mathcal{A}$ with $\mathcal{C}$ and requiring runs to satisfy the

acceptance condition of $\mathcal{A}$. Thus, $\mathcal{P} = \langle \Sigma, Q \times Q', \langle q_0, q_0' \rangle, \eta, \alpha \times \Delta' \rangle$, where for every state $\langle q, q' \rangle \in Q \times Q'$ and letter $\sigma \in \Sigma$, we have that $\eta(\langle q, q' \rangle, \sigma) = \delta(q, \sigma) \times \delta'(q', \sigma)$, and a transition $\langle \langle q, q' \rangle, \sigma, \langle p, p' \rangle \rangle$ is accepting iff the corresponding transition $\langle q, \sigma, q' \rangle$ of $\mathcal{A}$ is accepting. Since the transition functions of $\mathcal{A}$ and $\mathcal{C}$ are full, the product of $\mathcal{A}$ with $\mathcal{C}$ is also full and so $L(\mathcal{P}) = L(\mathcal{A})$. We prove that $\mathcal{P}$ can be pruned to an equivalent tDBW.

Consider the following game between Player $\exists$ and Player $\forall$. The game is played on $\mathcal{P}$ and starts from position $\langle q_0, q_0' \rangle$. When the game is in position $\langle q, q' \rangle \in Q \times Q'$, Player $\forall$ chooses a letter $\sigma \in \Sigma$, and Player $\exists$ chooses a successor position $\langle s, s' \rangle \in \eta(\langle q, q' \rangle, \sigma)$. The outcome ρ of an interaction between the players is an infinite sequence of Σ-labeled transitions of $\mathcal{P}$. That is, $\rho = (\langle q_0, q_0' \rangle, \sigma_1, \langle q_1, q_1' \rangle), (\langle q_1, q_1' \rangle, \sigma_2, \langle q_2, q_2' \rangle), (\langle q_2, q_2' \rangle, \sigma_3, \langle q_3, q_3' \rangle \ldots$. Note that ρ combines a run $r = \langle q_0, \sigma_1, q_1 \rangle, \langle q_1, \sigma_2, q_2 \rangle, \ldots$ of $\mathcal{A}$ with a run $r' = \langle q_0', \sigma_1, q_1' \rangle, \langle q_1', \sigma_2, q_2' \rangle, \ldots$ of $\mathcal{C}$, both on the word $w = \sigma_1 \cdot \sigma_2 \cdot \sigma_3 \cdots$ obtained by concatenating the letters chosen by Player $\forall$.

The winning condition for Player $\exists$ is that either r satisfies α or r' satisfies β. This winning condition can be specified by a transition-based parity condition with three colors. Thus, $\gamma : (Q \times Q') \times \Sigma \times (Q \times Q') \to \{0, 1, 2\}$ and an outcome ρ is winning for Player $\exists$ if $\max \gamma(\inf(\rho))$ is even.

For all $q, p \in Q$, $q', p' \in Q'$, and $\sigma \in \Sigma$ we define $\gamma(\langle q, q' \rangle, \sigma, \langle p, p' \rangle)$ as follows.

$$\gamma(\langle q, q' \rangle, \sigma, \langle p, p' \rangle) = \begin{cases} 0 \text{ if } \langle q, \sigma, p \rangle \notin \alpha \text{ and } \langle q', \sigma, p' \rangle \notin \beta \\ 1 \text{ if } \langle q, \sigma, p \rangle \notin \alpha \text{ and } \langle q', \sigma, p' \rangle \in \beta \\ 2 \text{ if } \langle q, \sigma, p \rangle \in \alpha \end{cases}$$

Outcomes ρ with $\max \gamma(\inf(\rho)) = 0$ are outcomes in which r' satisfies β, outcomes ρ with $\max \gamma(\inf(\rho)) = 2$ are outcomes in which r that satisfies α, and outcomes ρ with $\max \gamma(\inf(\rho)) = 1$ are such that r' does not satisfy β and r does not satisfy α. Note that r and r' are both runs above the same word in automata that complement each other, and so at most one of them is accepting. Since an outcome ρ is winning for Player $\exists$ if $\max \gamma(\inf(\rho)) \in \{0, 2\}$, the function γ specifies outcomes in which at least one of the runs are accepting.

It is easy to see that following the HD strategies of $\mathcal{A}$ and $\mathcal{C}$ is a winning strategy for Player $\exists$. Indeed, this strategy guarantees that if the word w is in $L(\mathcal{A})$, the run r is accepting in $\mathcal{A}$ and thus satisfies the Büchi condition α, namely infinitely many moves of priority 2 are played. Conversely, if $w \in \overline{L(\mathcal{A})}$, then while the run r is clearly rejecting, the run r' is accepting in $\mathcal{C}$ and thus eventually only moves of priority 0 are taken. Since every word is either in $L(\mathcal{A})$ or in $\overline{L(\mathcal{A})}$, the winning condition for Player $\exists$ is always satisfied.

It is known that parity games admit memoryless strategies and that a memoryless strategy can be computed in polynomial time for parity games with a fixed number of colors [11,10]. Hence, we can compute in polynomial time a memoryless winning strategy for Player $\exists$. Such a strategy maps each position $\langle q, q' \rangle \in Q \times Q'$ and letter $\sigma \in \Sigma$ to a position $\langle s, s' \rangle$, and induces the required pruning of $\mathcal{P}$ into a deterministic automaton. $\square$

Combining Theorems 2 and 3, we have the following.

Corollary 1. *Let $\mathcal{A}$ be an HD-tNBW with n states. There is a tDBW equivalent to $\mathcal{A}$ with at most n^2 states. Moreover, the determinization can be done in polynomial time.*

5 On the Size of HD strategies

The quadratic determinization construction for HD-tNBWs described in Theorem 1 implies a quadratic upper bound for the size of HD strategies. Indeed, given an HD-tNBW $\mathcal{A}$ with state space Q, the equivalent tDBW has state space $Q \times Q$, where a state $\langle q, q' \rangle$ can be viewed as a memory of the state q. In this section we show that the bound is tight – thus there are HD-tNBWs whose HD strategy must be quadratic.

Theorem 4. *For every $n \geq 1$, there is an HD-tNBW $\mathcal{A}_n$ such that $\mathcal{A}_n$ has $O(n)$ states and every HD strategy for $\mathcal{A}_n$ needs at least n^2 states.*

Proof. Consider $n \geq 3$. Let $[n] = \{1, 2, \ldots, n\}$. We define $\mathcal{A}_n = \langle [n] \cup \{\#\}, Q_n, a_1, \Delta_n, \alpha_n \rangle$ as follows (See $\mathcal{A}_3$ in Figure 1. For clarity, #-transitions are dashed in the figure and their # label is omitted).

- The state space $Q_n = \{a_i, b_i, c_i, d_i : i \in [n]\}$ is partitioned into four levels, each of size n. We refer to the states $a_1, \ldots, a_n$ as "the a-states" of $\mathcal{A}_n$, they constitute the first level of states, and similarly for b-, c-, and d-states, and the second, third, and fourth level, respectively.
- Transitions are only between adjacent levels, and Δ_n is the union of the following sets of transitions.
  - #-transitions from the first and third levels: $\{\langle a_i, \#, b_j \rangle : j \neq i \in [n]\} \cup \{\langle c_i, \#, d_i \rangle : i \in [n]\}$,
  - transitions from the second level: $\{\langle b_j, i, a_j \rangle : j \neq i \in [n]\} \cup \{\langle b_j, j, c_j \rangle : j \in [n]\}$, and
  - transitions from the fourth level: $\{\langle d_i, j, c_j \rangle : i, j \in [n]\}$.
- The set of accepting transitions (colored green in the figure) is $\alpha_n = \{\langle b_j, j, c_j \rangle : j \in [n]\} \cup \{\langle d_i, i, c_i \rangle : i \in [n]\}$. Thus, transitions from the second to the third level are accepting (note that these transitions are traversed at most once in each run, and so their membership in α is arbitrary), and reading i in state d_i, we go to c_i via an accepting transition.

Let L_n be the set of words w of the form $(\# \cdot [n])^\omega$ such that there is $i \in [n]$ such that the subword $\#i\#i$ appears infinitely often in w. We prove that $L(\mathcal{A}_n) = L_n$.

It is easy to see that $\mathcal{A}_n$ accepts only words of the form $(\# \cdot [n])^\omega$. It is less easy to see that it accepts a word $w \in (\# \cdot [n])^\omega$ iff there is $i \in [n]$ such that the subword $\#i\#i$ appears infinitely often in w. To see this, note that the only nondeterminizm in $\mathcal{A}_n$ is in the #-transitions from the first level. Once a run of $\mathcal{A}_n$ reaches the component that consists of the third and fourth levels, it stays

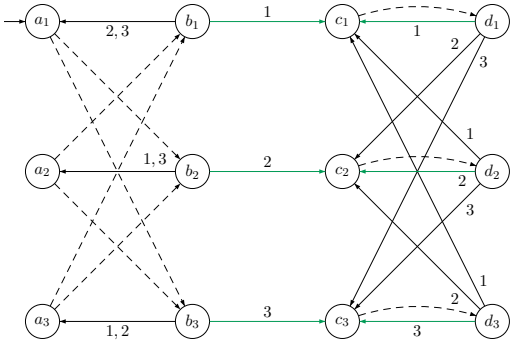


Fig. 1: The HD-tNBW $\mathcal{A}_3$.

there forever, it is deterministic, and it accepts only words in L_n. Thus, if a word is accepted, it is in L_n.

For the other direction, consider a word $w \in L_n$. Recall that the tNBW is deterministic in the limit, and its deterministic component recognizes L_n. So it will be enough to show that there is a run on w that enters the deterministic levels. In fact, we will prove that there is a strategy to generates a run that enters the deterministic levels for all $w \in L_n$. Assume by induction that we have generated on a prefix $u \in (\# \cdot [n])^*$ that ends with $i \in [k]$, a run that reached some a-state a_j. Thus by the time we finished reading u we have yet fulfilled our goal to reach a deterministic c-state. By definition of δ, if we go back from the b level to the a level with an i-transition, then it is to an a-state different from a_i. Thus $i \neq j$. Now if we move from a_j upon reading $\#$ to b_i, then in the next step we'll succeed and move to the third level iff the next letter will be i. Notice that if in the limit we always fail, then it means that we have never seen even a single good pattern in $\{\#k\#k : k \in [n]\}$. Since $w \in L_n$, it follows that at least one such pattern exists, and so after a finite amount of time the strategy enters the deterministic levels and from there the only deterministic run must be accepting.

The strategy described above requires each a-state to have $n-1$ memories – one for each letter with which the state was reached. Accordingly, as every a-state can be reached with $(n-1)$ letters, a finite-state transducer that models this strategy needs at least $n(n-1)$ memories for the first level, and then at least one memory for each of the other $3n$ states, so overall at least n^2+2n memories. We now prove that this is optimal. That is, that a minimal HD strategy for $\mathcal{A}_n$ needs at least n^2+2n memories. Note that this does not imply that every tDBW for L_n needs $\Omega(n^2)$ states. In particular, the deterministic component that consists of the third and fourth levels of $\mathcal{A}_n$ can serve as a linear tDBW for L_n.

Consider a minimal HD strategy $g = \langle [n] \cup \{\#\}, M, m_{init}, \rho, \tau \rangle$ for $\mathcal{A}_n$. We prove that each a-state a_i has at least $n-1$ memories in M, and hence by the

same analysis as above g has at least n^2+2n memories. Observe that if g has less than $n-1$ memories of some a-state a_i, then at least one of the non-deterministic #-transitions from a_i are not used, and hence $\mathcal{A}_n$ is not maximally trimmed. Indeed, Let $\mathcal{A}'_n$ be a tNBW obtained from $\mathcal{A}_n$ by trimming out transitions unused by g. On the one hand, $L(\mathcal{A}'_n) \subseteq L(\mathcal{A}_n)$, and on the other hand g witnesses that for all $w \in L(\mathcal{A}_n)$ the run generated by g on w is an accepting run also in $\mathcal{A}'_n$. So $L(\mathcal{A}_n) \subseteq L(\mathcal{A}'_n)$ and g is an HD strategy for $\mathcal{A}'_n$ as well.

Assume by way of contradiction that there is a transition #-transition from some a-state that can be removed from $\mathcal{A}_n$ while maintaining its language and HDness. W.l.o.g. assume that the transition $\langle a_1, \#, b_2\rangle$ can be removed, and let $h : ([n] \cup \{\#\})^* \to Q_n$ be a strategy that witnesses the HDness of the obtained tNBW.

Consider the word $w_1 = (\#1)^\omega$. Since $w_1 \in L_n$, it is accepted, and so there must be $i_1 \geq 0$ such that $h((\#1)^{i_1}\#) = b_1$. Indeed, the only way for a run to be accepted is to move from the second to the third level, which can be done only by reading the letter 1 from the state b_1. Let $x_1 = (\#1)^{i_1}\#2\#2$ and consider now the word $w_2 = x_1 \cdot (\#1)^\omega$. When a run of $\mathcal{A}_n$ on w_2 follows h, it reads the first occurrence of the letter 2 from state b_1. Hence, the run continues to the state a_1, and after reading the next #, it continues to a state b_j for some $j \neq 2$. Indeed, the transition $\langle a_1, \#, b_2\rangle$ does not exist. Hence, upon reading the next 2, it moves to the state a_j. As $w_2 \in L_n$, continuing to follow h eventually leads the run again to b_1. That is, there is $i_2 \geq 0$ such that $h(x_1 \cdot (\#1)^{i_2}\#) = b_1$. We let $x_2 = x_1 \cdot (\#1)^{i_2}\#2\#2$ and note again that when h reads x_2, it reaches a state b_j for some $j \neq 2$. We can now continue and define an infinite sequence of words $x_1, x_2, x_3, \ldots$, where each x_i is a proper prefix of x_{i+1} and contains at least i occurrences of the subword #2#2. In addition, for all $i \geq 1$ the run of $\mathcal{A}_n$ on x_i, when following h, never leaves the first and second levels. Thus, in the limit, we get a word with infinitely many occurrences of #2#2 for which the run of $\mathcal{A}_n$ generated by h is rejecting, and we reach a contradiction. □

Note that the automaton $\mathcal{A}_n$ used in the proof of Theorem 4 is *deterministic in the limit*: all accepting runs eventually reach a deterministic component. On the one hand, this makes the lower bound stronger, as it shows that the quadratic lower bound applies even for HD-tNBWs of a restricted class. In particular, note that the quadratic strategy is needed in order to make a single successful guess, which has the flavor of weak HD-tNBWs, which are known to be DBP [4]. On the other hand, it makes the result weaker, as by making one of the c-states initial, we get an equivalent tDBW. The latter concern can be easily addressed, by making the automaton $\mathcal{A}_n$ a bit more complicated by directing the accepting transitions from the fourth level back to the first level, namely defining $\delta(d_i, i) = a_i$ rather than $\delta(d_i, i) = c_i$. It is not hard to see that all the considerations made above for $\mathcal{A}_n$ apply also for this variant (which is the extension to n letters of the HD-NBW used in [3] in order to show than HD-NBWs need not be DBP).

References

1. B. Aminof, O. Kupferman, and R. Lampert. Reasoning about online algorithms with weighted automata. In *Proc. 20th ACM-SIAM Symp. on Discrete Algorithms*, pages 835–844, 2009.
2. M. Bagnol and D. Kuperberg. Büchi good-for-games automata are efficiently recognizable. In *Proc. 38th Conf. on Foundations of Software Technology and Theoretical Computer Science*, volume 122 of *LIPIcs*, pages 16:1–16:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.
3. U. Boker, D. Kuperberg, O. Kupferman, and M. Skrzypczak. Nondeterminism in the presence of a diverse or unknown future. In *Proc. 40th Int. Colloq. on Automata, Languages, and Programming*, volume 7966 of *Lecture Notes in Computer Science*, pages 89–100, 2013.
4. U. Boker, O. Kupferman, and M. Skrzypczak. How deterministic are Good-For-Games automata? In *Proc. 37th Conf. on Foundations of Software Technology and Theoretical Computer Science*, volume 93 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:14, 2017.
5. U. Boker and K. Lehtinen. When a little nondeterminism goes a long way: an introduction to history-determinism. *ACM SIGLOG News*, 10(1):177–196, 2023.
6. J.R. Büchi. On a decision method in restricted second order arithmetic. In *Proc. Int. Congress on Logic, Method, and Philosophy of Science. 1960*, pages 1–12. Stanford University Press, 1962.
7. T. Colcombet. The theory of stabilisation monoids and regular cost functions. In *Proc. 36th Int. Colloq. on Automata, Languages, and Programming*, volume 5556 of *Lecture Notes in Computer Science*, pages 139–150. Springer, 2009.
8. J. Esparza, J. Kretínský, and S. Sickert. One theorem to rule them all: A unified translation of LTL into ω-automata. In *Proc. 33rd IEEE Symp. on Logic in Computer Science*, pages 384–393, 2018.
9. T.A. Henzinger and N. Piterman. Solving games without determinization. In *Proc. 15th Annual Conf. of the European Association for Computer Science Logic*, volume 4207 of *Lecture Notes in Computer Science*, pages 394–410. Springer, 2006.
10. C.S. Jutla. Determinization and memoryless winning strategies. *Information and Computation*, 133(2):117–134, 1997.
11. N. Klarlund. Progress measures, immediate determinacy, and a subset construction for tree automata. In *Proc. 7th IEEE Symp. on Logic in Computer Science*, pages 382–393. IEEE Computer Society, 1992.
12. D. Kuperberg and M. Skrzypczak. On determinisation of good-for-games automata. In *Proc. 42nd Int. Colloq. on Automata, Languages, and Programming*, pages 299–310, 2015.
13. O. Kupferman. Automata theory and model checking. In *Handbook of Model Checking*, pages 107–151. Springer, 2018.
14. O. Kupferman. Using the past for resolving the future. *Frontiers in Computer Science*, 4, 2022.
15. O. Kupferman, S. Safra, and M.Y. Vardi. Relating word and tree automata. *Ann. Pure Appl. Logic*, 138(1-3):126–146, 2006.
16. O. Kupferman and M.Y. Vardi. Safraless decision procedures. In *Proc. 46th IEEE Symp. on Foundations of Computer Science*, pages 531–540, 2005.
17. L.H. Landweber. Decision problems for ω–automata. *Mathematical Systems Theory*, 3:376–384, 1969.

18. D. Niwinski and I. Walukiewicz. Relating hierarchies of word and tree automata. In *Proc. 15th Symp. on Theoretical Aspects of Computer Science*, volume 1373 of *Lecture Notes in Computer Science*. Springer, 1998.
19. M.O. Rabin and D. Scott. Finite automata and their decision problems. *IBM Journal of Research and Development*, 3:115–125, 1959.
20. S. Safra. On the complexity of ω-automata. In *Proc. 29th IEEE Symp. on Foundations of Computer Science*, pages 319–327, 1988.
21. M.Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994.