Past, Present, and Infinite Future

Thomas Wilke

Kiel University, Kiel, Germany
thomas.wilke@email.uni-kiel.de

Abstract

I was supposed to deliver one of the speeches at Wolfgang Thomas's retirement ceremony. Wolfgang had called me on the phone earlier and posed some questions about temporal logic, but I hadn't had good answers at the time. What I decided to do at the ceremony was to take up the conversation again and show how it could have evolved if only I had put more effort into answering his questions. Here is the imaginary conversation with Wolfgang.

The contributions are (1) the first direct translation from counter-free ω-automata into future temporal formulas, (2) a definition of bimachines for ω-words, (3) a translation from arbitrary temporal formulas (including both, future and past operators) into counter-free ω-bimachines, and (4) an automata-based proof of separation: every arbitrary temporal formula is equivalent to a boolean combination of pure future, present, and pure past formulas when interpreted in ω-words.

1998 ACM Subject Classification F.1.1 Models of Computation, F.1.2 Modes of Computation, F.4.1 Mathematical Logic, F.4.3 Formal Languages

Keywords and phrases linear-time temporal logic, separation, backward deterministic ω-automata, counter freeness

Digital Object Identifier 10.4230/LIPIcs.ICALP.2016.95

1 Act I

Wolfgang is sitting at his desk. Thomas is standing in his office, looking over the bay. They are talking to each other on the phone.

Wolfgang. I am teaching a course on applied automata theory this semester, and I would like to explain to my students how one can translate a counter-free ω-automaton into a temporal formula. I took a look at your STACS paper from 1999 [17], but the translation you give there is only for finite words. How does the whole story go for infinite words?

Thomas. I don't know of any published translation which comes close to what I present in the STACS paper. There is Volker and Paul's comprehensive contribution to the volume that celebrated your 60th birthday [4], but the construction presented therein is probably not what you are looking for, given its algebraic nature.

Wolfgang. I know what Volker and Paul did. Indeed, I am looking for something which is more automata-theoretic.

Thomas. I may have a suggestion for you.

Wolfgang. So?

Thomas. First of all, we need to choose the right ω-automaton model. Imagine you wanted to translate a future temporal formula over finite words into a finite-state automaton. Which model of automaton would you use?

Wolfgang. When I use ordinary automata, which read a word from left to right, I end up with a nondeterministic automaton, because the automaton can only guess what will happen in the future. When I use backward automata, which read a word from right to left, I end up with a deterministic automaton right away, simply because what happens in the future can easily be determined when coming back from it.

THOMAS. That's the point. When translating counter-free ω-automata into temporal formulas, we start out best from backward deterministic ω-automata, the model introduced by Olivier and Max Michel in their 1999 paper [3]. Do you recall how these automata are defined?

WOLFGANG. Sure, I do. I think Jean-Éric and Dominique call them "prophetic" automata in their book [11], but backward deterministic ω-automata is also a good term. Anyway, such an automaton is given by a finite state set Q, an initial recurrence condition $\mathcal{I}$, a backward transition function $\rho\colon \Sigma \times Q \to Q$, and a set $F \subseteq Q$ of final states. It is required that for every ω-word over Σ there is exactly one initial run.

THOMAS. Which recurrence conditions do you have in mind?

WOLFGANG. Nothing in particular. How about you?

THOMAS. Generalized transition Büchi conditions will come in handy. With such a condition, $\mathcal{I}$ is a set of sets $T \subseteq \Sigma \times Q$, each of them referred to as a transition recurrence set. A run $q_0q_1q_2\dots$ on a word $w \in \Sigma^\omega$ needs to satisfy $q_i = \rho(w(i), q_{i+1})$ for every $i < \omega$. For such a run to be initial it is required that for every transition recurrence set $T \in \mathcal{I}$ there are infinitely many i with $\langle w(i), q_{i+1}\rangle \in T$. It is final if $q_0 \in F$, and it is accepting if it is initial and final, just as usual, only that we are going the other direction.

WOLFGANG. OK. So we agree on the automaton model to be used.

THOMAS. The next thing we need to agree on is what "counter freeness" should mean for such automata.

WOLFGANG. To me, there seems to be a straightforward definition. For every finite word $w \in \Sigma^*$ we consider the function ${}^\rho w\colon Q \to Q$ induced by w on the state space. This is defined as usual, that is, ${}^\rho w(q) = {}^*\rho(w, q)$ for every $q \in Q$, where ${}^*\rho$ is the backward transition function extended from Σ to Σ^*. We say the automaton has a counter if there exists some word $w \in \Sigma^+$ and some nonempty subset $Q' \subseteq Q$ such that ${}^\rho w$ operates as a non-trivial permutation on Q'. More precisely, ${}^\rho w|_{Q'}$ is a bijection and ${}^\rho w|_{Q'} \neq \mathrm{id}_{Q'}$. If there is no counter, the automaton is counter-free.

THOMAS. You are perfectly right. There is no difference to what we know from finite words, and the definition coincides with Volker and Paul's definition for Büchi automata in general.

WOLFGANG. What I don't see right away is that the definition really captures the essence of counter freeness in general. I would like to understand this, but, maybe, it is better to return to this later.

THOMAS. Promised. – Let's start to work on a translation from counter-free backward deterministic ω-automata to future temporal formulas. I suggest we work with the usual vocabulary for temporal logic. For every symbol a in the alphabet, we have an atomic formula a, which is true in a word if the word starts with a. We may use the temporal operator "next", which we write as X, and the stutter-free "until", which we write as U. We may also use derived operators such as "eventually" and "always", which we write as F and G, respectively. Finally, boolean constants and operators are allowed as well.

WOLFGANG. I recall that your translation from counter-free automata to temporal logic is defined by induction, on the number of states of the given automaton in the first place and the size of the alphabet in the second place. Do we proceed in the same fashion for ω-words?

THOMAS. Yes, exactly. Recall that for every ω-word there is exactly one initial run of a given backward deterministic ω-automaton on this word. Let's call the first state of this run the halting state of the word. The inductive claim is that for every $q \in Q$, there exists a temporal formula defining the set of words with halting state q. – Wolfgang, things are getting more technical now. Let's use a desktop sharing software.

Thomas walks to his laptop. Wolfgang, already sitting in front of his screen, initiates a session between the two of them.

THOMAS. I was saying that for every $q \in Q$, we construct a temporal formula $\alpha[q, \rho, \mathcal{I}]$, which defines the set of ω-words with halting state q. For convenience, let's write $L_\omega(q, \rho, \mathcal{I})$ for this language.

WOLFGANG. I also recall you proceed by a simple case distinction. The almost trivial case is that all symbols induce the identity, which would mean ${}^\rho a = \mathrm{id}_Q$ for every $a \in \Sigma$.

THOMAS. We proceed by the same case distinction here. If all symbols induce the identity function, the translation is simple and does not need the inductive assumption, but it is slightly more complicated than for finite words, because we are using generalized transition Büchi conditions.

WOLFGANG. What exactly do you mean?

THOMAS. When every symbol induces the identity, then every run is of the form q^ω for some $q \in Q$. Whether or not such a run is initial for a given word depends on the symbols occurring infinitely often in the word.

WOLFGANG. That's funny. My first thought was that this case is really trivial.

THOMAS. It is almost trivial, because for every set $\Sigma' \subseteq \Sigma$ the formula

$$\bigwedge_{a \in \Sigma'} \mathrm{GF}a \wedge \bigwedge_{a \in \Sigma \setminus \Sigma'} \mathrm{FG}\neg a$$

specifies exactly the ω-words where Σ' is the set of symbols occurring infinitely often.

WOLFGANG. That's indeed almost trivial. *(Smiling.)* Let me understand what is going on in the more complicated case, when there is some symbol $c \in \Sigma$ such that ${}^\rho c$ is not the identity, that is, the image of ${}^\rho c$ is a strict subset of Q.

THOMAS. Let's say this image is Q', and let's refer to $\Sigma \setminus \{c\}$ by Γ.

WOLFGANG. In the proof for finite words, you split up each word in the positions where c occurs. If we do this here as well, there are three cases to distinguish: *i.* words which belong to Γ^ω, *ii.* words which belong to $\Sigma^* c \Gamma^\omega$, and *iii.* words which belong to $(\Gamma^* c)^\omega$.

THOMAS. That's exactly right. We can deal with these three cases separately, because if we have a formula for each of these cases, their disjunction is the formula we are looking for. Case *i* is almost straightforward. We restrict the given backward transition function ρ to the smaller alphabet Γ, say the result is ρ', and the induction hypothesis applies right away. We obtain a formula $\alpha[q, \rho', \mathcal{I}]$ for $L_\omega(q, \rho', \mathcal{I})$, and we can set

$$\alpha[q, \rho, \mathcal{I}] = \mathrm{G}\neg c \wedge \alpha[q, \rho', \mathcal{I}] \ .$$

In other words, we take what we get from the induction hypothesis and rule out the symbol c.

WOLFGANG. Why do we need to rule out c?

THOMAS. Here is a simple example. Assume our alphabet was $\{a, b, c\}$ and the formula we obtained by induction was the formula a. Then ac^ω would be a model of the formula a, but this word does not belong to Γ^ω.

WOLFGANG. Interesting, but I also have another question. Isn't it true that when we restrict ρ to ρ' as above we also need to restrict $\mathcal{I}$ appropriately, because in $\mathcal{I}$ there might be sets T with elements $\langle c, q \rangle$, but c does not belong to the underlying alphabet?

Thomas. Strictly speaking, you are right. That's why we should agree, once and for all, that we implicitly restrict recurrence conditions to the symbols occurring in the respective transition function.

Wolfgang. OK. – I think I know how to proceed in Case *ii.* Every word in $\Sigma^* c\Gamma^\omega$ can be broken up in a unique fashion. It can be written as ucv where $v \in \Gamma^\omega$. For the first part, u, we use what we know from finite words, and for the second part, v, we use the induction hypothesis.

Thomas. That's exactly right, but let's make it more precise. Let's write $L_*(q, \rho, q')$ for the language recognized by the backward deterministic automaton on finite words with initial state q', backward transition function ρ, and final state q. Then the language we are interested in, $L_\omega(q, \rho, \mathcal{I}) \cap \Sigma^* c\Gamma^\omega$, is the finite union of all languages

$$L_*(q, \rho, \rho(c, q'))\, c\, L_\omega(q', \rho', \mathcal{I}) \ , \tag{1}$$

for q' ranging over Q.

Wolfgang. Because of what we know about finite words, we have a temporal formula $\beta[q, \rho, q']$ for each language $L_*(q, \rho, q')$. And because of the induction hypothesis, we have a temporal formulas $\alpha[q', \rho', \mathcal{I}]$ for each language $L_\omega(q', \rho', \mathcal{I})$.

Thomas. Right! – A formula for a language as above, as given in (1), is therefore given by

$$\mathrm{xt}(\beta[q, \rho, \rho(c, q')]) \wedge \mathrm{F}(c \wedge \mathrm{XG}\neg c \wedge \mathrm{X}\alpha[q', \rho', \mathcal{I}]) \ ,$$

where $\mathrm{xt}(\beta)$ extends β to ω-words.

Wolfgang. I think I know what you mean by "extending" β to ω-words. You mean that for each $u \in \Sigma^*$ and $v \in \Gamma^\omega$, the word ucv is a model of $\mathrm{xt}(\beta)$ if, and only if, u is a model of β.

Thomas. You are perfectly right. – It is easy to obtain $\mathrm{xt}(\beta)$ from β by an inductive construction. The only interesting parts are the base case for a symbol and the induction steps involving temporal operators, because then the formula may "look" beyond the last occurrence of c, what is to be avoided:

$$\begin{aligned}
\mathrm{xt}(a) &= a \wedge \mathrm{XF}c \ , \\
\mathrm{xt}(\mathrm{X}\psi) &= \mathrm{X}(\mathrm{xt}(\psi) \wedge \mathrm{F}c) \ , \\
\mathrm{xt}(\psi_0 \mathrm{U} \psi_1) &= \mathrm{xt}(\psi_0)\mathrm{U}(\mathrm{xt}(\psi_1) \wedge \mathrm{F}c) \ .
\end{aligned}$$

Wolfgang. So the really interesting case is the last one, Case *iii*, when we consider words which belong to $(\Gamma^* c)^\omega$. Do you use an encoding trick like the one you use for finite words?

Thomas. Yes, the construction is very similar to the one in the finite-word setting, but considerably trickier. Let w be any word in $(\Gamma^* c)^\omega$. We can write w as $v_0 c v_1 c v_2 c \dots$ where $v_i \in \Gamma^*$ for every $i < \omega$. There is exactly one initial run for w, say $q_0 q_1 q_2 \dots$. Consider the subsequence $q_{i_0} q_{i_1} q_{i_2} \dots$ which collects the states the automaton assumes just left of any c.

Wolfgang. Because we assume the image of ${}^\rho c$ is Q', these states are special in the sense that each of them belongs to Q'. So in some sense, each of the cv_i's transforms some state from Q' to some state from Q'.

Thomas. That's it! – Basically, we classify each word in $c\Gamma^*$ according to how it operates on Q', more precisely, to every such word w we assign the function $\tilde{w} \colon Q' \to Q'$ defined by $\tilde{w}(q) = {}^*\rho(w, q)$ for every $q \in Q'$.

Wolfgang. I understand that, but what I am concerned about is that we lose information about the recurrence condition!

Thomas. That's why I said "basically". We do a little bit more. We not only assign $\tilde{w}$ to w but also $\dot{w}$, a function $Q' \to 2^{\mathcal{I}}$, which collects the information about the recurrence condition we need.

Wolfgang. I can guess how $\dot{w}$ is defined. For every state $q \in Q'$ we consider the backward run of A – viewed as a backward automaton on finite words – on w starting in q and collect in $\dot{w}(q)$ all transition recurrence sets it passes through.

Thomas. Exactly, that's how we do it. – The code alphabet, let's call it Δ, is large. For each finite word $w \in c\Gamma^*$ it contains the symbol $\langle \tilde{w}, \dot{w} \rangle$.

Wolfgang. How do we construct the backward deterministic ω-automaton, let's call it C, which uses this alphabet?

Thomas. Its state space is Q', which is smaller than the state space of A. Its backward transition function – let's denote it τ – is defined by $\tau(\langle f, g\rangle, q') = f(q')$ for every $q' \in Q'$. For each transition Büchi set $T \in \mathcal{I}$, the transition Büchi set $\mathcal{J}$ of C has a Büchi set T', which is given by $T' = \{\langle\langle f, g\rangle, q'\rangle \mid T \in g(q')\}$. Most importantly, C is a counter-free backward deterministic ω-automaton as defined above.

Wolfgang. I immediately see how C mimics A. Let's define a function $h_0 \colon c\Gamma^* \to \Delta$ by $h_0(w) = \langle \tilde{w}, \dot{w}\rangle$ and use this to define a function $h \colon (c\Gamma^*)^\omega \to \Delta^\omega$ by setting $h(cw_0cw_1\dots) = h_0(cw_0)h_0(cw_1)\dots$ for any choice of $w_i \in \Gamma^*$. Then, for every word $w \in (c\Gamma^*)^\omega$, the image $h(w)$ has halting state q' in C if, and only if, w has halting state q' in A. Or, formally, $L_\omega(q, \rho, \mathcal{I}) \cap (\Gamma^* c)^\omega$ is the finite union of all languages

$$L_*(q, \rho', q')\, h^{-1}(L_\omega(q', \tau, \mathcal{J})) \quad , \tag{2}$$

where q' ranges over all states in Q'.

Thomas. That's right! – Did you observe where the generalized transition Büchi condition came in handy?

Wolfgang. Yes, I did. If we had used some other condition, transferring it from A to C could have easily blown up the state space of C.

Thomas. So you see how we can use the induction hypothesis!?

Wolfgang. Sure. We can apply it to C, because C has a smaller state space than A, and obtain, for every $q' \in Q'$, a temporal formula $\alpha[q', \tau, \mathcal{J}]$ for $L_\omega(q', \tau, \mathcal{J})$, the alphabet being Δ.

Thomas. And do you see where the results from the finite-word setting come into the picture?

Wolfgang. Sure. They are used in two different places. First, for every $q \in Q$ and every $q' \in Q'$ there is a temporal formula $\beta[q, \rho, q']$ that defines $L_*(q, \rho', q')$. Second, for every symbol $\langle f, g\rangle \in \Delta$ there is a temporal formula $\chi_{f,g}$ such that a word $w \in \Gamma^*$ is a model of $\chi_{f,g}$ if, and only if, $h_0(cw) = \langle f, g\rangle$.

Thomas. Precisely! – What is left to be done is to assemble all these formulas in the right fashion.

Wolfgang. I can take over the first programming task. We need to transform every formula φ over Γ into a formula $\mathrm{xt}'(\varphi)$ over Σ such that for all $u \in \Gamma^*$ and $v \in \Sigma^\omega$, the formula φ is a model of u if, and only if, the formula $\mathrm{xt}'(\varphi)$ is a model of ucv. The crucial definitions are:

$$\begin{aligned} \mathrm{xt}'(a) &= a \ , \\ \mathrm{xt}'(\mathrm{X}\psi) &= \neg c \wedge \mathrm{X}\,\mathrm{xt}'(\psi) \ , \\ \mathrm{xt}'(\psi_0 \mathrm{U} \psi_1) &= (\mathrm{xt}'(\varphi) \wedge \neg c) \mathrm{U}\, \mathrm{xt}'(\psi) \ . \end{aligned}$$

THOMAS. Second, we need to transform every formula φ over Δ into a corresponding formula $\mathrm{lft}(\varphi)$, which "lifts" the formula from the alphabet Δ to the alphabet Σ, more precisely, for each $u \in (c\Sigma^*)^\omega$, the word u is a model of $\mathrm{lft}(\varphi)$ if, and only if, $h(u)$ is a model of φ. Here, we can set:

$$\begin{aligned} \mathrm{lft}(\langle f, g\rangle) &= \mathrm{xt}'(\chi_{f,g}) \ , \\ \mathrm{lft}(\mathrm{X}\varphi) &= \mathrm{X}(\neg c \mathrm{U}(c \wedge \mathrm{X}\,\mathrm{lft}(\varphi))) \ , \\ \mathrm{lft}(\varphi \mathrm{U} \psi) &= (c \rightarrow \mathrm{X}\,\mathrm{lft}(\varphi))\mathrm{U}(c \wedge \mathrm{X}\,\mathrm{lft}(\psi)) \ . \end{aligned}$$

WOLFGANG. We are done. The overall formula for a language as in (2) is

$$\mathrm{GF}c \wedge \mathrm{xt}'(\beta[q, \rho', q']) \wedge \neg c \mathrm{U}(c \wedge \mathrm{lft}(\alpha)[q', \tau, \mathcal{J}]) \ .$$

Thomas, I need to run. There's a meeting I have to attend ...

Wolfgang grabs a pile of documents from a bookshelf and leaves his room in a hurry.

2 Act II

Wolfgang, wearing a headset, and Thomas are sitting at their desks. On Thomas's screen, a notification from the desktop sharing software pops up. Wolfgang is trying to connect. Thomas puts on his headset.

THOMAS. Wolfgang?

WOLFGANG. Thomas! Can I come back to the discussion on temporal logic and counter-free automata we has the other day?

THOMAS. Sure. What is it that you would like to talk about?

WOLFGANG. You said that your definition of "counter-free ω-automaton" is a good one; you even said it would be the right one. Can you explain that to me?

THOMAS. From finite words we know that there are many equivalent definitions of what it means for a formal language to be star-free: recognizable by a counter-free automaton [13], definable in first-order logic [10], definable in temporal logic (with future and past operators) [8], definable in future temporal logic [7], ... For ω-words, the same equivalences hold, in particular, first-order logic, the two variants of temporal logic, and star-free expressions are equally expressive [9, 15, 8, 7]. Do you recall this? *(Smiling.)* Anyway, any notion of "counter freeness" that is equivalent to one of these formalisms should be ok.

WOLFGANG. I agree! – What we already know is that every counter-free automaton according to your definition is equivalent to a future temporal logic formula. So if you can also prove the converse to me, I will be happy.

THOMAS. To tell you the truth, parts of this were already proved a long time ago, but went unnoticed. What I mean is that the straightforward translation of a future temporal formula into a generalized Büchi automaton yields a counter-free backward deterministic

ω-automaton. In fact, for all I know, this observation was the motivation for defining backward deterministic ω-automata [1] and you can find the translation, for instance, in a paper by Olivier [2]. – The problem is to show that this translation yields a counter-free automaton.

WOLFGANG. *(Indignant, but smiling.)* Thomas! – That we obtain backward deterministic Büchi automata when translating future temporal formulas into automata is something I have known for years, in fact, I explained this to one of the anonymous referees of this paper already 15 years ago. But I didn't look at counter freeness at the time I have to admit. So let's see what happens when we follow a construction like the one Pierre and Moshe suggested [16]?

THOMAS. A typical automaton for a temporal formula guesses, for each point in time, which subformulas are true at that point and which are not.

WOLFGANG. So we model a state as a function $f\colon \mathrm{sbf}(\varphi) \to \{0,1\}$, where $\mathrm{sbf}(\varphi)$ stands for the set of subformulas of φ. The functions considered are required to satisfy the following straightforward conditions: $f(\top) = 1$; $f(\neg\psi) = 1$ if, and only if, $f(\psi) = 0$; $f(\psi_0 \vee \psi_1) = 1$ if, and only if, $f(\psi_0) = 1$ or $f(\psi_1) = 1$. Here, ψ and $\psi_0 \vee \psi_1$ stand for elements of $\mathrm{sbf}(\varphi)$. The transition relation, let's denote it by Δ, is defined according to the semantics of temporal logic, more precisely, $(f, a, g) \in \Delta$ if, and only if, the following conditions are satisfied:

- $f(a) = 1$,
- $f(b) = 0$ for every symbol $b \in \Sigma$ with $b \neq a$,
- $f(\mathrm{X}\psi) = g(\psi)$,
- $f(\psi_0 \mathrm{U} \psi_1) = 1$ if, and only if, $f(\psi_1) = 1$ or, $f(\psi_0) = 1$ and $g(\psi_0 \mathrm{U} \psi_1) = 1$.

Just as above, $\mathrm{X}\psi$ and $\psi_0 \mathrm{U} \psi_1$ stand for elements of $\mathrm{sbf}(\varphi)$. And, of course, the transition relation Δ is backward deterministic, that is, $\rho(a, g) = f$ for $(f, a, g) \in \Delta$ is well-defined.

THOMAS. What about the recurrence condition?

WOLFGANG. We need to make sure that when a U-formula is guessed to be true it becomes true eventually. To this end, we use a generalized state Büchi recurrence condition, which can easily be transformed into a generalized transition Büchi condition. For every U-subformula, say $\psi_0 \mathrm{U} \psi_1$, we have the set

$$\{f \mid f(\psi_0 \mathrm{U} \psi_1) = 0 \text{ or } f(\psi_1) = 1\}$$

as an element of the recurrence condition. – So how do we know this automaton is counter-free?

THOMAS. This needs a proof indeed. – Suppose $Q' \subseteq Q$ is a nonempty subset of the state space and ${}^{\rho}w$ restricted to Q' is a permutation for some word $w \in \Sigma^+$. We want to show ${}^{\rho}w(f) = f$ for every state $f \in Q'$. One way to do this is to fix an element $f_0 \in Q'$ and consider the orbit of f_0.

WOLFGANG. What do you mean by "orbit"?

THOMAS. Define f_{i+1} by $f_{i+1} = {}^{\rho}w(f_i)$ for $i < \omega$. Then, because we assume ${}^{\rho}w$ restricted to Q' is a permutation, $f_m = f_0$ for some $m > 0$. The set $\{f_m \mid i < m\}$ is what we call the orbit of f_0.

WOLFGANG. I see. If we can prove $f_1 = f_0$, or, equivalently, $f_i = f_{i'}$ for all $i, i' < m$, we are done, because this means ${}^{\rho}w(f_0) = f_0$.

THOMAS. Exactly. By induction, we show $f_i(\psi) = f_{i'}(\psi)$ for every $\psi \in \mathrm{sbf}(\varphi)$ and all $i, i' < m$, which is sufficient. In fact, we prove something stronger.

WOLFGANG. I have no idea what that could be.

THOMAS. We refine the picture in an adequate fashion by introducing more states and making it cyclic. Let $n = |w|$. First, we extend w in both directions by repeating it, that is, we consider the sequence $(a_i)_{i\in\mathbf{Z}}$ defined by $a_{i+kn} = w(i)$ for all $i < n$ and $k \in \mathbf{Z}$ with $k \neq 0$. Second, we define, for every $i \in \mathbf{Z}$, a state g_i by

$$\begin{aligned} g_0 &= f_0 \ , \\ g_i &= {}^{\rho}a_i(g_{i+1}) && \text{for } i \text{ with } -mn < i < 0 \ , \\ g_{i+kmn} &= g_i && \text{for } i \text{ with } -mn < i \leq 0 \text{ and } k \neq 0 \ . \end{aligned}$$

Then $g_{-ni} = f_i$ for every $i \in \mathbf{N}$ and $g_i = {}^{\rho}a_i(g_{i+1})$ for all $i \in \mathbf{Z}$. Furthermore, $g_i(\psi) = g_{i'}(\psi)$ for all $i, i' \in \mathbf{N}$ with $i \equiv i'\ (mn)$ and $\psi \in \text{sbf}(\varphi)$. I write $i \equiv i'\ (l)$ to denote the fact that i and i' are identical after reduction modulo l. The stronger claim is that $g_i(\psi) = g_{i'}(\psi)$ holds for all $i, i' \in \mathbf{Z}$ with $i \equiv i'\ (n)$ and $\psi \in \text{sbf}(\varphi)$.

Wolfgang. I see, this contains the original claim as a special case. – I am curious to see how the inductive proof goes.

Thomas. For $\psi = \top$, the claim is trivial. For the other base case, assume $a \in \Sigma$ and $\psi = a$. We have $g_i(a) = 1$ if, and only if, $a = a_i$, which proves the claim, because if $i \equiv i'\ (n)$, then $a_i = a_{i'}$, by definition of $(a_i)_{i\in\mathbf{Z}}$.

Wolfgang. Now it's my turn. First, the claim is trivial for the boolean connectives $\neg$ and $\vee$. Assume $\mathrm{X}\psi \in \text{sbf}(\varphi)$. Then $g_i(\mathrm{X}\psi) = g_{i+1}(\psi)$ for every $i \in \mathbf{Z}$, because of the definition of Δ. By induction hypothesis, we have $g_{i+1}(\psi) = g_{i'+1}(\psi)$ for all i and i' with $i \equiv i'\ (n)$, which then implies the claim.

Thomas. Let me conclude the proof by looking at the case where $\psi_0 \mathrm{U} \psi_1 \in \text{sbf}(\varphi)$. It is good to proceed by a case distinction.

Wolfgang. I can imagine what the cases are.

Thomas. The first case is when $g_i(\psi_0) = 1$ holds for all $i \in \mathbf{Z}$. If $g_{i_0}(\psi_0 \mathrm{U} \psi_1) = 1$ for some i_0, then, by definition of Δ, $g_i(\psi_0 \mathrm{U} \psi_1) = 1$ for all $i \leq i_0$. Since $g_{i_0}(\psi_0 \mathrm{U} \psi_1) = 1$ means $g_{i_0+kmn}(\psi_0 \mathrm{U} \psi_1) = 1$ for all $k \in \mathbf{N}$, we even have $g_i(\psi_0 \mathrm{U} \psi_1) = 1$ for all $i \in \mathbf{Z}$.

The second case is when $g_{i_0}(\psi_0) = 0$ holds for some $i_0 \in \mathbf{Z}$. In this case, $g_{i_0+kmn}(\psi_0) = 0$ for all $k \in \mathbf{Z}$. So if $g_i(\psi_0 \mathrm{U} \psi_1) = 1$ for some $i \in \mathbf{Z}$, then, because of the definition of Δ, there exists some $l \in \mathbf{N}$ with $g_{i+l}(\psi_1) = 1$ and $g_{i+k}(\psi_0) = 1$ for all k with $k < l$ – a simple proof by induction shows that. Let i' be such that $i \equiv i'\ (n)$. Then $i + j \equiv i' + j\ (n)$ for every $j \in \mathbf{Z}$. So from the induction hypothesis, we can conclude $g_{i'+l}(\psi_1) = 1$ and $g_{i'+k}(\psi_0) = 1$ for all k with $k < l$. Hence, $g_{i'}(\psi_0 \mathrm{U} \psi_1) = 1$.

Wolfgang. We are done. Perfect!

Thomas. Wolfgang, my theory lecture starts in a few minutes. I hope you don't mind me hanging up now. I am calling back later.

Wolfgang. I am sorry. Please, go ahead.

Thomas grabs his tablet, a copy of Sipser's book [14], and leaves his room; Wolfgang takes a notebook and starts scribbling.

3 Act III

Wolfgang is still sitting at his desk, contemplating. Thomas just entered his office, carrying his tablet and Sipser's book, and went straight to his laptop. He is initiating another session with Wolfgang.

Wolfgang. Thomas, thanks for calling back. I hope your lecture went well. Here is what I thought about in the meantime. Now that we know how to deal with counter-free ω-automata and future temporal logic, can we also say something about temporal logic in general? When future *and* past operators are allowed?

Thomas. Why are you asking? – We somehow know what happens because of the result by Gabbay, Pnueli, Shelah, and Stavi [7].

Wolfgang. I know this result settles the question in the sense that every arbitrary temporal formula is equivalent to a future temporal formula when interpreted in the first position of ω-words. I also know that by a result by Gabbay [6] every arbitrary temporal formula is equivalent to a boolean combination of pure past, present, and pure future formulas. But proofs of these results are technically involved and I am missing automata theory in this picture!

Thomas. Oh, that's an interesting thought.

Thomas thinking …

Thomas. I think we can say something really nice here. Again, we need to agree on the right automaton model.

Wolfgang. What are you thinking of?

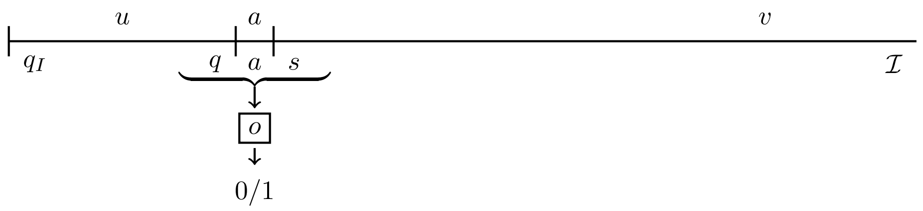
Thomas. Well, arbitrary temporal logic formulas are interpreted in some position in an ω-word. In some sense, the semantics of such a formula is a function $[\![\varphi]\!]$ which maps ω-words over Σ to ω-words over $\{0,1\}$, where the bit at position i of the image of a given word is the truth value of the formula when interpreted at position i in the given word. So what I think we should do is to come up with a slick automaton model for describing such functions.

Wolfgang. There are many ways to define functions from ω-words to ω-words and even more ways to define functions from finite words to finite words. A very general notion is that of a rational function and it has been studied in detail. For instance, there is a result by Elgot and Mezei which states that a rational function of finite words is the composition of a left-sequential and a right-sequential function [5], and I believe I have come across a similar result for ω-words.

Thomas. That is true. Elgot and Mezei's result was generalized to ω-words by Olivier [2]. I am thinking of extending Schützenberger's bimachines [12] to ω-words in a way adequate for dealing with temporal logic.

Wolfgang. I remember Schützenberger's bimachines. How do we extend them to ω-words?

Thomas. Suppose we want to realize a function $\nu\colon \Sigma^\omega \to \Gamma^\omega$. Suppose we are given a word $w \in \Sigma^\omega$. And suppose we want to know the symbol at position i in $\nu(w)$, that is, we want to know $\nu(w)(i)$. What we could do first is to split w at position i, that is, write w as uav where the length of u is i. Then we could run a forward deterministic automaton on u, a backward deterministic ω-automaton on v, and output $\nu(w)(i)$ depending on the two halting states and the symbol a. – Here is a picture. *(Drawing on the screen.)*



Wolfgang. That is very close to what Schützenberger suggested for finite words, only that he allowed an arbitrary word to be output, rather than a single symbol. – Let me try to make your picture formal and define what we may call ω-bimachines. Such a machine, let's call it A, consists of

- a forward deterministic automaton without final condition, say with finite state set Q, initial state q_I, and transition function $\delta\colon Q \times \Sigma \to Q$,
- a backward deterministic ω-automaton without final condition, say with finite state set S, initial recurrence condition $\mathcal{I}$, and a backward transition function $\rho\colon \Sigma \times S \to S$, and
- an output function $o\colon Q \times \Sigma \times S \to \Gamma$.

THOMAS. That's it. The semantics is a function $\Sigma^\omega \to \Gamma^\omega$, which we denote by ν. Assume $w \in \Sigma^\omega$ and we want to define $\nu(w)(i)$ for some $i \in \omega$. We take the halting state of the forward automaton for the word $w(0)\dots w(i-1)$, say this is q. We take the halting state of the backward automaton for the word $w(i+1)w(i+2)\dots$, say this is s. Then $\nu(w)(i) = o(q, w(i), s)$.

WOLFGANG. That is indeed a very simple and intuitive definition. It also connects nicely with rational functions, because from Olivier's results it should follow that the class of functions computed by ω-bimachines is the same as the class of total letter-to-letter rational functions. – What we want to do is to transform every temporal formula into an equivalent ω-bimachine.

THOMAS. We should even strive for a counter-free ω-bimachine, where this simply means that the forward and the backward automaton are counter-free. Because if we manage to achieve that, we also have a proof of the result by Gabbay, saying that every arbitrary temporal formula is equivalent to a boolean combination of pure future, present, and pure past formulas – completely in automata-theoretic terms.

WOLFGANG. That sounds like an interesting plan and I would like to give it a shot. In the future-only setting, it was easy to describe the state space in one go – it was a function $\mathrm{sbf}(\varphi) \to \{0,1\}$. I believe it is going to be more complicated here, which is the reason I suggest we try an inductive definition.

THOMAS. I will be happy with an inductive definition!

WOLFGANG. There are two base cases: $\psi = \top$ and $\psi = a$ for some $a \in \Sigma$. In both cases, we can choose the forward and the backward automaton to be a 1-state automaton. In the first case, we can set $o(q, a, s) = 1$ for every $a \in \Sigma$; in the second case, we can set $o(q, a, s) = 1$ and $o(q, b, s) = 0$ for all $b \in \Sigma \setminus \{a\}$. – Easy!

THOMAS. Clearly, these automata are counter-free.

WOLFGANG. In the inductive step, we have to take care of boolean operators and temporal operators.

THOMAS. Boolean operators can be dealt with easily, only the temporal operators are interesting.

WOLFGANG. We have four temporal operators when we admit future as well as past operators and follow standard syntax and semantics: X – next, P – previously, U – until, and S – since. I suggest we consider "previously" and "until", the two other can be dealt with in a similar fashion.

THOMAS. What we need for each of the two operators are two things. First, we need a construction. Second, we need a proof that it preserves counter freeness.

WOLFGANG. So let's turn to "previously" and assume we are given a counter-free ω-bimachine which computes a function $\mu\colon \Sigma^\omega \to \{0,1\}^\omega$. We want to construct a new counter-free ω-bimachine which computes the function ν defined by $\nu(w)(i+1) = \mu(w)(i)$ and $\nu(w)(0) = 0$ for every $i < \omega$. Apparently, we don't have to change the backward automaton; we only need to adapt the forward automaton and the output function.

THOMAS. I agree. This seems to be a simple construction.

WOLFGANG. We make the forward automaton lag behind in the sense that it keeps track of its previous state and the symbol just read. In the beginning, we start from a new state. So the state space is $Q \times \Sigma \cup \{\bot\}$ and the transition function is given by $\delta'(\langle q, a\rangle, b) = \langle \delta(q, a), b\rangle$ and $\delta'(\bot, a) = \langle q_I, a\rangle$. The output function, o', is given by $o'(\langle q, a\rangle, b, s) = o(q, a, \rho(b, s))$ and $o'(\bot, a, s) = 0$.

Thomas. This is quite convincing. In fact, the construction does not only seem to be correct to me. It also preserves counter freeness, as far as I can see.

Wolfgang. So let's turn to "until".

Thomas. We assume we are given two counter-free ω-bimachines computing functions $\mu_0 \colon \Sigma^\omega \to \{0,1\}^\omega$ and $\mu_1 \colon \Sigma^\omega \to \{0,1\}^\omega$, respectively, and we want to construct a counter-free ω-bimachine computing the function $\nu \colon \Sigma^\omega \to \{0,1\}^\omega$ given by $\nu(w)(i) = 1$ if, and only if, there exists some $k \geq i$ such that $\mu_1(w)(k) = 1$ and $\mu_0(w)(j) = 1$ for all j with $i \leq j < k$. – By taking a product of the two forward automata and the two backward automata, we can simplify the situation in the sense that we can think of only one counter-free ω-bimachine but with two output functions, o_0 and o_1, where the first one is for μ_0 and the second one is for μ_1. – Do you see what we need to do?

Wolfgang. Yes, right away. Let's call the given automaton A and the one to be constructed A'. It is important that the backward automaton of A' knows, at any point, those states from the forward automaton of A from which the U-formula can be satisfied. – I am aware of the fact that this is a vague description, but it should become clear when we work out the details.

Thomas. Let me see if I understand what you mean. The state space of the backward automaton of A' is $S \times 2^Q$. Its backward transition function is defined by

$$\rho'(a, \langle s, P\rangle) = \langle \rho(a, s), P'\rangle \ ,$$

where P stands for a subset of Q and P' is defined by

$$P' = \{q \in Q \mid o_1(q, a, s) = 1\} \cup \{q \in Q \mid \delta(q, a) \in P \text{ and } o_0(q, a, s) = 1\} \ .$$

The forward automaton of A' is the same as the one of A. The output function of A', let's denote it o', is given by $o'(q, a, \langle s, P\rangle) = 1$ if, and only if, $o_1(q, a, s) = 1$, or $o_0(q, a, s) = 1$ and $\delta(q, a) \in P$.

Wolfgang. This is what I was thinking of. The transition function ρ' reflects the semantics of the until operator in a particular sense, which I would like to make precise. For every state $q \in Q$, let A_q denote the ω-bimachine which is obtained from A by changing the initial state of the forward automaton to q. Further, let ν_0^q and ν_1^q denote the corresponding functions. Now suppose $w \in \Sigma^\omega$ and $\langle s_0, P_0\rangle\langle s_1, P_1\rangle \dots$ is a run of the backward automaton of A' on w such that $s_0 s_1 \dots$ is an initial run of the backward automaton of A and write P for P_0. Then the following is true for every state $q \in Q$ and can be proved by a straightforward induction:

1. If there is some j such that $\nu_1^q(j) = 1$ and $\nu_0^q(i) = 1$ for every $i < j$, then $q \in P$.

2. If $q \in P$, then either
  a. there is some j such that $\nu_1^q(j) = 1$ and $\nu_0^q(i) = 1$ for every $i < j$, or
  b. $\nu_0^q(i) = 1$ for all $i < \omega$.

The problem I see is that we really want 2.a for each $q \in P$. For a U-formula to be true, it is not enough to have 2.b only.

Thomas. Your concern is completely valid. Without any further measure, the construction may "overapproximate". The problem is similar to the fairness problem we had when we looked at the backward deterministic ω-automaton for a given future temporal formula and introduced a recurrence set for every U-subformula.

Wolfgang. What do you suggest? Are we going to do the same here?

Thomas. The situation is more complicated. Here is the basic idea. For every q satisfying 2.a there is a smallest j with the specified property. Let's denote this by j_q. We assign a natural number $p(q)$ to every state $q \in P$ such that $p(q)$ reflects the order of the j_q's, that is, $p(q) < p(q')$ if, and only if, $j_q < j_{q'}$. We then make sure by appropriate recurrence conditions that the numbers that are assigned to the successors of q decrease over time until j_q is finally reached.

Wolfgang. That sounds interesting. Can you make this precise?

Thomas. Let $m = |Q|$ and set $M = \{0, \ldots, m-1, \infty\}$. A state of the backward automaton of A' is then a pair $\langle s, p \colon Q \to M \rangle$ where P from above is now given by $\{q \in Q \mid p(q) \neq \infty\}$.

Wolfgang. Given this, I think I can describe how the correct backward deterministic transition function works. In a first step, we set

- $p_0(q) = -1$ for every q with $o_1(q, a, s) = 1$,
- $p_0(q) = p(\delta(q, a))$ for every q with $o_0(q, a, s) = 1$ and $o_1(q, a, s) = 0$, and
- $p_0(q) = \infty$ for all other $q \in Q$.

This is consistent with the definition of P' from above in the sense that $P' = \{q \in Q \mid p(q) \neq \infty\}$, but there are values out of range – we may have -1 as a value. We adjust p_0 to obtain p' by increasing some of its values as follows, where t is defined by $t = \min\{j \in \{-1, \ldots, m-1\} \mid p_0^{-1}(j) = \emptyset\}$.

- $p'(q) = p_0(q) + 1$ for all q with $p(q) < t$.
- $p'(q) = p_0(q)$ for all other $q \in Q$.

In some sense, we are adjusting as little as is necessary.

Thomas. This is absolutely correct. What we still have to define is an appropriate recurrence condition. This will be the union of two recurrence sets $\mathcal{I}_0$ and $\mathcal{I}_1$, where $\mathcal{I}_0$ simply extends $\mathcal{I}$ to the new state space in a straightforward way. More precisely, $\mathcal{I}_0 = \{S' \times M^Q \mid S' \in \mathcal{I}\}$.

The set $\mathcal{I}_1$ contains a transition recurrence set T_i for every $i \in \{0, \ldots, m-1\}$. This set contains a pair $\langle a, \langle s, p \rangle \rangle$ if, and only if, $t = i$ for the value t as defined above or $p'(q) \in \{0, \ldots, i-1, \infty\}$ for every $q \in Q$ and p' as defined above.

Wolfgang. Thomas, I understand what you are saying, but I think a rigorous correctness proof is needed here.

Thomas. I agree. As the construction is inspired by Oliver and Max Michel's work, we can also borrow some of their ideas for the correctness proof.

Wolfgang. Anyway, we also need to prove that our construction really yields a counter-free automaton.

Thomas. I agree again. The proof of this is tedious, but doable. One can use what we know from the proof that the backward deterministic ω-automaton for a given future temporal formula is counter-free.

Wolfgang. OK. There is still some work to be done before I can present the material to the students. I am calling it a day. Thanks a lot, and see you soon, Thomas!

Thomas. Bye, bye. And, please, say "hello" to Renate.

Wolfgang and Thomas close their desktop sharing applications.

4 Epilogue

A few weeks later, Thomas receives the following email.

Hi, Thomas!

You probably want to know how it went with my lectures. First of all, it didn't take me much time to work out all the details of what we talked about. Then everything went fine, only the students came up with a fair number of questions I couldn't answer right away. Here are the most interesting ones, maybe:

- What is the complexity of the translation from counter-free backward deterministic ω-automata to future temporal formulas?
- What is the complexity of the translation from arbitrary temporal formulas to ω-bimachines? Non-elementary?
- Say "prop" is some interesting class of deterministic automata. What happens when we consider "prop" ω-bimachines instead of counter-free ω-bimachines?

Any ideas?

All the best, Wolfgang

References

1 Olivier Carton. Personal communication.

2 Olivier Carton. Right-sequential functions on infinite words. In Farid M. Ablayev and Ernst W. Mayr, editors, *Computer Science – Theory and Applications, 5th International Computer Science Symposium in Russia, CSR 2010, Kazan, Russia, June 16-20, 2010. Proceedings*, volume 6072 of *Lecture Notes in Computer Science*, pages 96–106. Springer, 2010. doi:10.1007/978-3-642-13182-0_9.

3 Olivier Carton and Max Michel. Unambiguous Büchi automata. In Gaston H. Gonnet, Daniel Panario, and Alfredo Viola, editors, *LATIN 2000: Theoretical Informatics, 4th Latin American Symposium, Punta del Este, Uruguay, April 10-14, 2000, Proceedings*, volume 1776 of *Lecture Notes in Computer Science*, pages 407–416. Springer, 2000. doi:10.1007/10719839_40.

4 Volker Diekert and Paul Gastin. First-order definable languages. In Jörg Flum, Erich Grädel, and Thomas Wilke, editors, *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]*, volume 2 of *Texts in Logic and Games*, pages 261–306. Amsterdam University Press, 2008.

5 Calvin C. Elgot and Jorge E. Mezei. On relations defined by generalized finite automata. *IBM Journal of Research and Development*, 9(1):47–68, Jan 1965. doi:10.1147/rd.91.0047.

6 Dov M. Gabbay. The declarative past and imperative future: Executable temporal logic for interactive systems. In Behnam Banieqbal, Howard Barringer, and Amir Pnueli, editors, *Temporal Logic in Specification, Altrincham, UK, April 8-10, 1987, Proceedings*, volume 398 of *Lecture Notes in Computer Science*, pages 409–448. Springer, 1987.

7 Dov M. Gabbay, Amir Pnueli, Saharon Shelah, and Jonathan Stavi. On the temporal basis of fairness. In Paul W. Abrahams, Richard J. Lipton, and Stephen R. Bourne, editors, *Conference Record of the Seventh Annual ACM Symposium on Principles of Programming Languages, Las Vegas, Nevada, USA, January 1980*, pages 163–173. ACM Press, 1980. URL: http://dl.acm.org/citation.cfm?id=567446, doi:10.1145/567446.567462.

8 Johan Anthony Willem Kamp. *Tense logic and the theory of linear order*. PhD thesis, University of California, Los Angeles, 1968.

9 Richard E. Ladner. Application of model theoretic games to discrete linear orders and finite automata. *Information and Control*, 33(4):281–303, 1977. doi:10.1016/S0019-9958(77)90443-0.

10 Robert McNaughton and Seymour Papert. *Counter-free automata.* M.I.T. Press research monographs. M.I.T. Press, 1971.

11 Dominique Perrin and Jean-Éric Pin. *Infinite Words: Automata, Semigroups, Logic, and Games*, volume 141 of *Pure and Applied Mathematics.* Elsevier, Amsterdam, 2004.

12 Marcel Paul Schützenberger. A remark on finite transducers. *Information and Control*, 4(2-3):185–196, 1961. doi:10.1016/S0019-9958(61)80006-5.

13 Marcel Paul Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8(2):190–194, 1965. doi:10.1016/S0019-9958(65)90108-7.

14 Michael Sipser. *Introduction to the Theory of Computation.* Cengage Learning, Boston, Mass., 3rd edition, 2013.

15 Wolfgang Thomas. Star-free regular sets of omega-sequences. *Information and Control*, 42(2):148–156, 1979. doi:10.1016/S0019-9958(79)90629-6.

16 Moshe Y. Vardi and Pierre Wolper. Reasoning about infinite computations. *Inf. Comput.*, 115(1):1–37, 1994. doi:10.1006/inco.1994.1092.

17 Thomas Wilke. Classifying discrete temporal properties. In Christoph Meinel and Sophie Tison, editors, *STACS 99, 16th Annual Symposium on Theoretical Aspects of Computer Science, Trier, Germany, March 4-6, 1999, Proceedings*, volume 1563 of *Lecture Notes in Computer Science*, pages 32–46. Springer, 1999. doi:10.1007/3-540-49116-3_3.