Near-Optimal Scheduler Synthesis for LTL with Future Discounting

Shota Nakagawa and Ichiro Hasuo

University of Tokyo, Japan

Abstract. We study synthesis of optimal schedulers for the linear temporal logic (LTL) with future discounting. The logic, introduced by Almagor, Boker and Kupferman, is a quantitative variant of LTL in which an event in the far future has only discounted contribution to a truth value (that is a real number in the unit interval [0, 1]). The precise problem we study—it naturally arises e.g. in search for a scheduler that recovers from an internal error state as soon as possible—is the following: given a Kripke frame, a formula and a number in [0, 1] called a margin, find a path of the Kripke frame that is optimal with respect to the formula up to the prescribed margin (a truly optimal path may not exist). We present an algorithm for the problem: it relies on a translation to quantitative automata and their optimal value problem, a technique that is potentially useful also in other settings of optimality synthesis.

1 Introduction

In the field of formal methods where a mathematical approach is taken to modeling and verifying systems, the conventional theory is built around the Boolean notion of truth: if a given system satisfies a given specification, or not. This qualitative theory has produced an endless list of notable achievements from hardware design to communication protocols. Among many techniques, automata-based ones for verification and synthesis have been particularly successful in serving engineering needs, by offering a specification method by temporal logic and push button-style algorithms. See e.g. [15, 18].

However, trends today in the use of computers—namely, computers as part of more and more heterogeneous systems—have pushed researchers to turn to quantitative consideration of systems, too. For example, in an embedded system where a microcomputer controls a bigger system with mechanical/electronic components, concerns include real-time properties—if an expected task is finished within the prescribed deadline—and resource consumption e.g. with respect to electricity, memory, etc.

Quantities in formal methods can thus arise from a specification (or an objective) that is quantitative in nature. Another source of quantities are systems that are themselves quantitative, such as one with faulty components whose mathematical model is a probabilistic transition structure.

Besides, quantities in formal methods can arise simply via refinement of the Boolean notion of satisfaction. For example, the usual interpretation of the linear temporal logic (LTL) formula $F\varphi$—it is satisfied by a sequence $s_0s_1\ldots$ if there exists $i$ such that $s_i \models \varphi$—has the following natural quantitative refinement: the truth value $\llbracket s_0s_1\ldots, F\varphi \rrbracket \in$
[0, 1] of $s_0s_1\ldots$ is defined by

$$[s_0s_1\ldots, F\varphi] = (\frac{1}{2})^i,$$

where $i$ is the least index such that $s_i \models \varphi$.

This value $[s_0s_1\ldots, F\varphi] \in [0, 1]$ is a quantitative truth value and is like utility in the game-theoretic terminology. Such refinements allow quantitative reasoning about so-called quality of service (QoS), specifically “how soon $\varphi$ becomes true” in this example.

Another example is a quantitative variation of $G\varphi$, where $[s_0s_1\ldots, G\varphi] = 1 - (\frac{1}{2})^i$—where $i$ is the least index such that $s_i \not\models \varphi$—meaning that violation of $\varphi$ in the far future only has a small negative impact.

**LTL$^\text{disc}[D]$: LTL with Future Discounting** The last examples are about quantitative refinement of temporal specifications. An important step in this direction is taken in the recent work \cite{2} by Almagor, Boker and Kupferman. There various useful quantitative refinements in LTL—including the last examples—are unified under the notion of future discounting, an idea first presented in \cite{9} in the field of formal methods. They introduce a clean syntax of the logic LTL$^\text{disc}[D]$—called LTL with discounting—that has a “discounting until” operator $U_\eta$ as well as the non-discounting one $U$; they define its semantics; and importantly, they show that usual automata-theoretic techniques for verification and synthesis (e.g. from \cite{15,18}) mostly remain applicable.

Probably the most important algorithm in \cite{2} is for the threshold model-checking problem: given a Kripke structure $K$, a formula $\varphi$ and a threshold $v \in [0, 1]$, it asks if $[K, \varphi] > v$, i.e. the worst case truth value of a path of $K$ is above $v$ or not.\footnote{Another notable result in \cite{2} shows undecidability of the validity problem of LTL with future discounting, when the logic additionally has the propositional average operator $\oplus$. See also \cite{5}.}

The core idea of the algorithm is what we call an event horizon: assuming that a discounting function $\eta$ in $U_\eta$ tends to 0 as time goes by, and that $v > 0$, there exists a time beyond which nothing is significant enough to change the answer to the threshold model-checking problem. In this case we can approximate an infinite path by its finite prefix.

**Our Contribution: Near-Optimal Scheduler Synthesis for LTL$^\text{disc}[D]$** Now that a temporal formula $\varphi$ assigns quantitative truth or utility $[\xi, \varphi]$ to each path $\xi$, a natural task is to find a path $\xi_0$ in a given Kripke structure $K$ that achieves the optimal. We expect this problem has numerous instances in various application areas: a simple example is given by $K$ modeling a disk drive and a specification $\varphi = G(err \rightarrow F_\eta \text{recov})$, where $F_\eta$ is discounting by a suitable function $\eta$ and $G$ is not. By finding an optimal path $\xi_0$ in $K$, we synthesize a scheduler of the disk drive that recovers from an internal error as soon as possible. Moreover, from the value $[\xi_0, \varphi]$, we can read off the worst case error-recovery latency of the disk drive $K$.

It turns out, however, that a (truly) optimal path need not exist (Example 4.4): $v_0 = \sup_{\xi \in \text{path}(K)} [\xi, \varphi]$ is obviously a limit point but no $\xi_0$ achieves $[\xi_0, \varphi] = v_0$. This leads us to the following near-optimal scheduler synthesis problem:

**Near-optimal scheduler synthesis.** Given a Kripke structure $K$, an LTL$^\text{disc}[D]$ formula $\varphi$ and a margin $\varepsilon \in (0, 1)$, find a path $\xi_0 \in \text{path}(K)$ that is $\varepsilon$-optimal, that is, $\sup_{\xi \in \text{path}(K)} [\xi, \varphi] - \varepsilon \leq [\xi_0, \varphi]$.

Our main contribution is an automata-theoretic algorithm for this problem. It is PSPACE in the description lengths of $\varphi$ and $\varepsilon$, and NLOGSPACE in the size of $K$. As
usual the algorithm constructs from $\varphi$ and $\varepsilon$ an automaton $A_{\varphi, \varepsilon}$ with which we combine $K$. Running a nonemptiness check-like algorithm to the resulting automaton yields an answer. A major technical challenge was the definition of $A_{\varphi, \varepsilon}$: it must take sequences of discount factors into account, and is therefore substantially more complicated than in [2] (they still have the same singly exponential complexity).

On the one hand, the algorithm resembles the one in [2]. In particular it relies on the idea of event horizon: a margin $\varepsilon$ in our setting plays the role of a threshold in [2] and enables us to ignore events in the far future.

On the other hand, a major difference from [2] is that we translate a specification $(\varphi, \varepsilon)$ into an automaton that is itself quantitative (a $[0, 1]$-acceptance automaton, with Boolean branching and $[0, 1]$-acceptance values). This is unlike [2] where the target automaton is totally Boolean. An advantage of $[0, 1]$-acceptance automata is that they allow optimal path search much like emptiness of Büchi automata is checked (via lasso computations). Applied to our current problem, this enables us to directly synthesize a near-optimal path for $\text{LTL}^{\text{disc}}[\mathcal{D}]$ without knowing the optimal value $\sup_{\xi \in \text{path}(K)} [\xi, \varphi]$.

We speculate that this method of translation to $[0, 1]$-acceptance automata be useful in many other problems about quantitative optimization of dynamic systems. It would play the fundamental role that is played by some well-known techniques in other settings, namely: emptiness check for Büchi automata, via lasso computations, in the qualitative world; and computation of reachability probability in MC or MDP, via BSCCs, in probabilistic systems (the current method can be seen as an instance of the latter).

As additional material, we study the relationship between $[0, 1]$-acceptance automata and fuzzy automata [10, 16] (Appendix B). They recognize the same class of languages.

**Related Work** Quantitative variations of temporal logics and their decision procedures have been a very active research topic [1, 2, 5, 9, 11]. We shall lay them out along a basic taxonomy. We denote by $K$ (the model of) the system against which a specification formula $\varphi$ is verified (or tested, synthesized, etc.).

- **Quantitative vs. Boolean system models.** Sometimes we need quantitative considerations just because the system $K$ itself is quantitative. This is the case e.g. when $K$ is a Markov chain, a Markov decision process, a timed or hybrid automaton, etc. In the current work $K$ is a Kripke structure and is Boolean.

- **Quantitative vs. Boolean truth values.** The previous distinction is quite orthogonal to whether a formula $\varphi$ has truth values from $[0, 1]$ (or another continuous domain), or from $\{\text{tt, ff}\}$. For example, the temporal logic PCTL [12] for reasoning about probabilistic systems has modalities like $P_{>\nu}\psi$ (“$\psi$ with a probability $> \nu$”) and has Boolean interpretation. In $\text{LTL}^{\text{disc}}[\mathcal{D}]$ studied here, truth values are from $[0, 1]$.

- **Linear time vs. branching time.** This distinction is already there in the qualitative/Boolean setting [17]—its probabilistic variant is studied in [8]—and gives rise to temporal logics with the corresponding flavors (LTL vs. CTL, CTL*). In fact the idea of future discounting is first introduced to a branching-time logic in [9], where an approximation algorithm for truth values is presented.

- **Propositional vs. temporal quantitative operators.** In this paper we focus on quantitative connectives that are temporal: non-Boolean truth values arise only from future discounting. In contrast, propositional quantitative operators include: $\nabla_\lambda \psi$ (“multiply the truth value of $\psi$ by $\lambda \in (0, 1)$”); and $\psi_1 \oplus \psi_2$ (“take the average of
the truth values”). The work [1] focuses on propositional quantitative operators and shows that, with temporal ones absent, many known Boolean decision procedures can be adapted to their quantitative setting. In contrast, adding $\oplus$ to LTL$^\text{disc}[D]$ (with future discounting) leads to undecidability of validity [2].

- **Future discounting vs. future averaging.** The temporal quantitative operators in LTL$^\text{disc}[D]$ are discounting—an event’s significance tends to 0 as time proceeds—a fact that benefits model checking via event horizons. Different temporal quantitative operators are studied in [5], including the long-run average operator $\tilde{G}\psi$. Presence of $G$, however, makes most common decision problems undecidable [5].

In [11], LTL (without additional quantitative operators) is interpreted over the unit interval $[0,1]$, and its model-checking problem against quantitative systems $\mathcal{K}$ is shown to be decidable. In this setting—where the LTL connectives are interpreted by idempotent operators $\min$ and $\max$—the variety of truth values arises only from a finite-state quantitative system $\mathcal{K}$, hence is finite.

In [2, Thm. 4] it is proved that the threshold synthesis problem for LTL$^\text{disc}[D]$ is feasible. This problem asks: given a partition of atomic propositions into the input and output signals, an LTL$^\text{disc}[D]$ formula $\varphi$ and $v \in [0,1]$, to come up with a transducer (i.e. a finite-state strategy) that makes the truth value of $\varphi$ at least $v$. We remark that this is different from the near-optimal scheduler synthesis problem that we solve in this paper. The synthesis problem in [1, §2.2], without a threshold, is closer to ours.

Automata- (or game-) theoretic approaches are taken in [4, 6] to the synthesis of controllers or programs with better quantitative performance, too. In these papers, a specification is given itself as an automaton, as opposed to a temporal formula in the current work. Another difference is that, in [4, 6], utility is computed along a path by limit-averaging, as opposed to future discounting here. The algorithms in [4,6] therefore rely on those which are known for mean-payoff games, including the ones in [7].

More and more diverse quantitative measures of systems’ QoS are studied recently: from best/worst case probabilities and costs, to quantiles, conditional probabilities and ratios. See [3] and the references therein. Study of such in LTL$^\text{disc}[D]$ is future work.

**Organization of the Paper** In [2] we review the logic LTL$^\text{disc}[D]$ and known results on threshold model checking and satisfiability, all from [2]. We introduce quantitative variants of (alternating) Büchi automata, called (alternating) $[0,1]$-acceptance automata, in [3] with auxiliary observations on their relation to fuzzy automata [16]. These automata play a central role in [4] where we formalize and solve the near-optimal scheduler synthesis problem for LTL$^\text{disc}[D]$. Complexities are studied in [4] in §6 we conclude, citing some future work. Omitted proofs are found in Appendix A.

## 2 LTL$^\text{disc}[D]$ and Threshold Problems

We review the linear temporal logic LTL$^\text{disc}[D]$ introduced in [2].

**Definition 2.1 (discounting function)** A discounting function is a strictly decreasing function $\eta : \mathbb{N} \to [0,1]$ such that $\lim_{i \to \infty} \eta(i) = 0$. A special case is an exponential discounting function $\exp_\lambda$, where $\lambda \in (0,1)$, that is defined by $\exp_\lambda(i) = \lambda^i$.

The set $E = \{\exp_\lambda \mid \lambda \in (0,1) \cap \mathbb{Q}\}$ is that of exponential discounting functions.
Remark 2.2 An extension of the framework is proposed in [2] where a discounting function $\eta$ need not tend to 0. It is claimed that such an extension, e.g. when $\eta$ tends to $\frac{1}{2}$, is suited for the situation where we are (not totally pessimistic but) ambivalent about the future. This extension does not change the algorithmic results in [2], nor here. It is enough that the limit value $\lim_{k\to\infty} \eta(k)$ is statically known so that we can use the value in construction of automata.

The logic $\text{LTL}^{\text{disc}}[D]$ is parametrized by the set $D$ of discounting functions, a fact that becomes relevant in the analysis of complexity. See [5] where we fix $D = E$.

Definition 2.3 (LTL$^{\text{disc}}[D]$) We fix the set $AP$ of atomic propositions. Given a set $D$ of discounting functions, the formulas of LTL$^{\text{disc}}[D]$ are defined by the grammar:

$$\varphi ::= \text{True} \mid p \mid \neg \varphi \mid \varphi \land \varphi \mid X\varphi \mid \varphi U \varphi \mid \varphi U_\eta \varphi,$$

where $p \in AP$ and $\eta \in D$ is a discounting function. We adopt the usual notation conventions: $F\varphi = \text{True} U \varphi$ and $G\varphi = \neg F \neg \varphi$. The same goes for discounting operators: $F_\eta \varphi = \text{True} U_\eta \varphi$ and $G_\eta \varphi = \neg F_\eta \neg \varphi$.

Definition 2.4 (semantics of LTL$^{\text{disc}}[D]$, [2]) Let $\pi = \pi_0\pi_1\ldots \in (P(AP))^\omega$ be a computation, and $\varphi$ be an LTL$^{\text{disc}}[D]$ formula. The truth value $[\pi, \varphi]$ of $\varphi$ in $\pi$ is a real number in $[0, 1]$ defined as follows. Here $\pi^i = \pi_i\pi_{i+1}\ldots$ is a suffix of $\pi$.

$$[\pi, \text{True}] = 1 \quad [\pi, \neg \varphi] = 1 - [\pi, \varphi] \quad [\pi, \varphi_1 \land \varphi_2] = \min\{[\pi, \varphi_1], [\pi, \varphi_2]\}$$

$$[\pi, X\varphi] = [\pi^1, \varphi]$$

$$[\pi, \varphi_1 U \varphi_2] = \sup_{i \in \mathbb{N}} \{\min\{[\pi^i, \varphi_1], \min_{0 \leq j < i}[\pi^j, \varphi_1]\}\}$$

$$[\pi, \varphi_1 U_\eta \varphi_2] = \sup_{i \in \mathbb{N}} \{\min\{\eta(i)[\pi^i, \varphi_2], \min_{0 \leq j < i}[\eta(j)[\pi^j, \varphi_1]]\}\}$$

Lemma 2.5 The truth value $[\pi, \varphi_1 U \varphi_2]$ lies between 0 and $\eta(0)$. □

We extend the semantics to Kripke structures.

Definition 2.6 Let $AP$ be the set of atomic propositions. A Kripke structure is a tuple $\mathcal{K} = (W, R, \lambda)$, where $W$ is a finite set of states, $R \subseteq W^2$ is a transition relation and $\lambda : Q \to P(AP)$ is a labeling function.

A path of a Kripke structure $\mathcal{K}$ is an infinite sequence $\xi = s_0s_1\ldots$ of states $s_i \in W$ such that $(s_i, s_{i+1}) \in R$ for each $i \in \mathbb{N}$. The set of paths of $\mathcal{K}$ is denoted by $\text{path}(\mathcal{K})$.

Definition 2.7 Let $\mathcal{K}$ be a Kripke structure and $\xi$ be a path of $\mathcal{K}$. The truth value $[\xi, \varphi]$ of $\varphi$ in $\xi$ is defined by $[\xi, \varphi] = [\lambda(\xi), \varphi]$ where $\lambda : W \to P(AP)$ is extended to a function $W^\omega \to (P(AP))^\omega$ from paths to computations.

The truth value $[\mathcal{K}, \varphi]$ of $\varphi$ in $\mathcal{K}$ is defined by $[\mathcal{K}, \varphi] = \inf_{\xi \in \text{path}(\mathcal{K})} [\xi, \varphi]$.

It is shown in [2] that the following “threshold” problems are decidable. Their complexities are studied in [2], too, restricting to $D = E$ (Def. 2.4). The complexities are expressed in terms of a suitable notion $|\langle \varphi \rangle|$ of the size of $\varphi$ (see 3).
Theorem 2.8 (2) The threshold model-checking problem for \( LTL^{disc}[D] \) is: given a Kripke structure \( K \), an \( LTL^{disc}[D] \) formula \( \varphi \) and a threshold \( v \in [0, 1] \), decide whether \( \llbracket K, \varphi \rrbracket \geq v \). It is decidable; when restricted to \( LTL^{disc}[E] \) and \( v \in \mathbb{Q} \), the problem is in \( \text{PSPACE} \) in \( |\llbracket \varphi \rrbracket| \) and in the description of \( v \), and in \( \text{NLOGSPACE} \) in the size of \( K \).

The threshold satisfiability problem for \( LTL^{disc}[D] \) is: given an \( LTL^{disc}[D] \) formula \( \varphi \), a threshold \( v \in [0, 1] \) and \( \sim \in \{<, >\} \), decide whether there exists a computation \( \pi \in (P(AP))^\omega \) such that \( \llbracket \pi, \varphi \rrbracket \sim v \). This is decidable; when restricted to \( LTL^{disc}[E] \) and \( v \in \mathbb{Q} \), the problem is in \( \text{PSPACE} \) in \( |\llbracket \varphi \rrbracket| \) and in the description of \( v \). \hfill \qed

3 [0, 1]-Acceptance Büchi Automata

Our algorithm for near-optimal scheduler synthesis relies on a certain notion of quantitative automaton—called \([0, 1]\)-acceptance Büchi automaton, see Def. 3.1—and an algorithm for its optimal value problem (Lem. 3.2). The notion may well be known, but to the best of our knowledge it is not extensively studied in the literature.

In a \([0, 1]\)-acceptance Büchi automaton each state has a real value \( v \in [0, 1] \), instead of a Boolean value \( b \in \{\text{tt, ff}\} \), of acceptance. Note that branching is Boolean (i.e. nondeterministic) and not \([0, 1]\]-weighted. In Appendix B we show that adding weights to branching does not increase expressivity when it comes to (weighted) languages.

Definition 3.1 ([0, 1]-acceptance automaton) A \([0, 1]\)-acceptance Büchi automaton—or simply a \([0, 1]\)-acceptance automaton henceforth—is a tuple \( \mathcal{A} = (\Sigma, Q, I, \delta, F) \), where \( \Sigma \) is a finite alphabet, \( Q \) is a finite set of states, \( I \subseteq Q \) is a set of initial states, \( \delta : Q \times \Sigma \to \mathcal{P}(Q) \) is a transition function and \( F : Q \to [0, 1] \) is a function that assigns an acceptance value to each state. We define the language \( \mathcal{L}(\mathcal{A}) : \Sigma^\omega \to [0, 1] \) of \( \mathcal{A} \) by

\[
\mathcal{L}(\mathcal{A})(w) = \max\{F(q) | \exists \rho \in \text{run}(w). q \in \text{Inf}(\rho)\}.
\]

Here the notion of \( \text{run} \) is defined as usual, and \( \text{Inf} \) denotes the set of state which occurs infinitely many times, that is, \( \text{Inf}(\rho) = \{q \in Q | \forall n \in \mathbb{N}. \exists m > n. q = \rho_m\} \).

The following observation, though not hard, is a key fact for our synthesis algorithm. It is a quantitative analogue of emptiness check in usual (Boolean) automata.

Lemma 3.2 (the optimal value problem for [0, 1]-acceptance automata) Let \( \mathcal{A} = (\Sigma, Q, I, \delta, F) \) be a \([0, 1]\)-acceptance Büchi automaton. There exists the maximum \( \max_{w \in \Sigma^\omega} \mathcal{L}(\mathcal{A})(w) \) of \( \mathcal{L}(\mathcal{A}) \). Moreover, there is an algorithm that computes the value \( \max_{w \in \Sigma^\omega} \mathcal{L}(\mathcal{A})(w) \) as well as a run \( \rho_{\max} = q_0a_0q_1a_1 \ldots \in (\Sigma \times Q)^\omega \) that realizes the maximum.

Proof. The algorithm is much like the one for emptiness check of (ordinary) Büchi automata, searching for a suitable lasso computation. More concretely: consider those states which are both reachable from an initial state and reachable from itself. Let \( s \) be one with the greatest acceptance value \( F(s) \). It is easy to show that a lasso computation with the state \( s \) as a “knot” gives the run \( \rho_{\max} \) that we seek for. \hfill \qed
are interested in the following problem: what path of a given Kripke structure is best for a given LTL formula \( \phi \)?

In Appendix B we show that this generalization does not add expressivity. In fact, the set of positive propositional formulas (using \( \land, \lor, \top, \bot \)) over \( q \in Q \) and \( v \in [0, 1] \) as atomic propositions.

We define the language \( \mathcal{L}(A) : \Sigma^\omega \to [0, 1] \) of \( A \) by

\[
\mathcal{L}(A)(w) = \max_{\tau \in \text{run}(w)} \min_{\rho \in \text{path}(\tau)} F^\infty(\rho),
\]

where runs, paths and the function \( F^\infty \) are defined as follows. The notion of run here is much like for the usual alternating automata: a run \( \tau \), given an alternating automaton \( A \) and a word \( w \in \Sigma^\omega \), is a possibly infinite-depth tree (whose branching is thought of as conjunction). Then a path \( \rho \) of such a run is either: 1) an infinite sequence \( q_0 q_1 \ldots \) of states; or 2) a finite sequence \( q_0 q_1 \ldots q_n v \), with \( q_i \in Q \) and \( v \in [0, 1] \). The function \( F^\infty \) in (1) is then defined by

\[
F^\infty(q_0 q_1 \ldots) = \max_{q \in \text{inf}(q_0 q_1 \ldots)} F(q), \quad \text{and} \quad F^\infty(q_0 q_1 \ldots q_n v) = v.
\]

Lemma 3.4 Let \( A = (\Sigma, Q, I, \delta, F) \) be an alternating \([0, 1]\)-acceptance automaton. There exists a \([0, 1]\)-acceptance automaton \( A' \) such that \( \mathcal{L}(A) = \mathcal{L}(A') \).

The construction of \( A' \) is a quantitative adaptation of the one [14] that turns an alternating \( \omega \)-automaton into nondeterministic. In our adaptation we use what we call exposition flags, an idea that is potentially useful in other settings with Büchi-type acceptance conditions, too. The proof, as well as some further optimization, is in Appendix A.1.

A generalization of \([0, 1]\)-acceptance automaton is naturally obtained by making transitions also \([0, 1]\)-weighted. The result is called fuzzy automaton and studied e.g. in [16]. In Appendix B we show that this generalization does not add expressivity. In fact we prove a more general result, parametrizing \([0, 1]\) into a suitable semiring \( \mathbb{K} \).

4 Near-Optimal Scheduler Synthesis for LTL\(^{\text{disc}}[D] \)

In [2] the threshold model-checking problem for LTL\(^{\text{disc}}[D] \) is studied. In this paper we are interested in the following problem: what path of a given Kripke structure \( \mathcal{K} \) is the best for a given LTL\(^{\text{disc}}[D] \) formula \( \phi \).

In general, however, there does not exist an optimal path \( \xi_0 \) of \( \mathcal{K} \), i.e. one that achieves \( [\xi_0, \phi] = \sup_{\xi \in \text{path}(\mathcal{K})} [\xi, \phi] \).

Example 4.1 (optimality not achievable) Take a formula \( \phi = G_p F \) and the Kripke structure shown in the above. We note that, in each path \( \xi \) of the Kripke structure, \( p \) is

\[\begin{align*}
\mathcal{K} & : \mathcal{K} = (S, \mathcal{D}, \mathcal{D}, p, \xi_0, \xi_1, \xi_2) \\
S & = \{s_0, s_1, s_2\}, \quad \mathcal{D} = \{p, \neg p\}, \\
\mathcal{D} & = \{p, \neg p\}, \\
p & = \xi_0 \xrightarrow{p} \xi_1, \quad \xi_1 \xrightarrow{\neg p} \xi_2 \\
\end{align*}\]
true at most once. The later such a state occurs in a path $\xi$, the bigger the truth value $[\xi, \varphi]$ is; moreover the value $[\xi, \varphi]$ converges to 1 (since $\eta$ converges to 0). However there is no path $\xi$ that achieves exactly $[\xi, \varphi] = 1$: if $p$ is postponed indefinitely, no state in $\xi$ satisfies $p$, in which case $Fp$ is everywhere false and hence $[\xi, \varphi] = 0$.

We thus strive for near-optimality, allowing a prescribed margin $\varepsilon$.

**Definition 4.2** The near-optimal path synthesis problem for $\text{LTL}^{\text{disc}}[D]$ is: given a Kripke structure $\mathcal{K} = (W, R, \lambda)$, an $\text{LTL}^{\text{disc}}[D]$ formula $\varphi$ and a positive real number $\varepsilon \in (0, 1)$, find a path $\xi_0 \in \text{path}(\mathcal{K})$ such that $[\xi_0, \varphi] \geq \sup_{\xi \in \text{path}(\mathcal{K})} [\xi, \varphi] - \varepsilon$.

Our algorithm for this problem first translates $\varphi$ and $\varepsilon$ to an alternating $[0, 1]$-acceptance automaton $\mathcal{A}_{\varphi, \varepsilon}$. It is further transformed to a $[0, 1]$-acceptance automaton (Lem. 3.4). The resulting automaton—after taking the product with $\mathcal{K}$—is amenable to optimal value search (Lem. 4.2), yielding a solution to the original problem.

### 4.1 The Alternating $[0, 1]$-Acceptance Automaton $\mathcal{A}_{\varphi, \varepsilon}$

Here describe the construction of $\mathcal{A}_{\varphi, \varepsilon}$. After presenting its (rather complicated) definition, we discuss ideas behind it, comparing the definition with other known constructions. A lemma follows (Lem. 4.7) that formulates the correctness of the construction.

We recall some notions from [2].

**Definition 4.3** ($\eta^+ k$, $\text{xcl}(\varphi)$) Let $\eta : \mathbb{N} \rightarrow [0, 1]$ be a discounting function. We define a discounting function $\eta^+ k : \mathbb{N} \rightarrow [0, 1]$ by $\eta^+ k(i) = \eta(i + k)$ for each $k \in \mathbb{N}$.

For an $\text{LTL}^{\text{disc}}[D]$ formula $\varphi$, the extended closure $\text{xcl}(\varphi)$ of $\varphi$ [2] is defined by

$$\text{xcl}(\varphi) = \text{Sub}(\varphi) \cup \{ \varphi_1 \cup_{\eta^+ k} \varphi_2 \mid k \in \mathbb{N}, \varphi_1 \cup_0 \varphi_2 \in \text{Sub}(\varphi) \} ,$$

where $\text{Sub}(\varphi)$ denotes the set of subformulas of $\varphi$.

In the alternating $[0, 1]$-acceptance automaton $\mathcal{A}_{\varphi, \varepsilon}$ that we shall construct, a state is a pair $(\psi, \mathbf{d})$ of a formula $\psi$ and a discount sequence $\mathbf{d} \in [0, 1]^+$, the latter being a sequence of real numbers that are thought of as discount factors. We organize them in a sequence $\mathbf{d} = d_1 \cdot d_2 \ldots d_n$, rather than simply multiplying them into a single number $d_1 \cdot d_2 \ldots d_n \in [0, 1]$, in order to keep track of the alternation between fixed point operators with different “polarities” (least or greatest). For example, the formula $F_{\eta_1} G_{\eta_2} F_{\eta_3} p$ will induce a discount sequence $\langle \eta_1(n_1), \eta_2(n_2), \eta_3(n_3) \rangle$ of length 3, where $n_1, n_2$ and $n_3$ are the numbers of steps for which $F_{\eta_1}$, $G_{\eta_2}$ and $F_{\eta_3}$ “have waited,” respectively.

We define two operators $\odot, \boxtimes$ that involve discount sequences. We use $\mathbf{d} \boxtimes v$ is the truth value $v$ “discounted by” $\mathbf{d}$. Further explanations will follow.

**Definition 4.4** (discount sequence: $\odot, \boxtimes$) A discount sequence is a sequence $\mathbf{d} = d_1 \cdot d_2 \ldots d_n \in [0, 1]^+$ of real numbers with a nonzero length. The update operator $\odot$ takes a discount
sequence \( d' \) and a discount factor \( d'' \in [0,1] \) as arguments, and the outcome is the sequence with the last element of \( d'' \) multiplied by \( d' \). That is,

\[
(d_1d_2\ldots d_n) \odot d' = d_1d_2\ldots d_{n-1}(d_n \cdot d') \in [0,1]^+.
\]

The action operator \( \boxdot \) takes \( d' \in [0,1]^+ \) and \( v \in [0,1] \) as arguments. The value \( d' \boxdot v \in [0,1] \) is defined inductively by:

\[
(d_1d_2\ldots d_n) \boxdot v = d_1 - d_2d_3 - \ldots + (-1)^n d_1d_2\ldots d_{n-1} + (-1)^n+1 d_nv.
\]

The intuition behind the action \( d' \boxdot v \) is most visible in (3). Given a discount sequence \( d''d' \), what happens is: 1) we apply the last discount factor \( d'' \) to the truth value \( v \), obtaining \( d' \cdot v \); 2) the concatenation of \( d'' \) to \( d' \) means there was a change of polarities, leading to the negation \( 1 - d' \cdot v \) (cf. Def. \( \Sigma,\Omega \)); and finally 3) we apply the remaining sequence \( d'' \) and obtain \( d'' \boxdot (1 - d' \cdot v) \). We note a straightforward relationship between \( \odot \) and \( \boxdot \):

\[
(d'' \circ d') \boxdot v = d'' \boxdot (d' \cdot v).
\]

We turn back to the construction of \( A_{\phi,\varepsilon} \). We first define \( A_{\phi,\varepsilon} \) that is infinite-state, and obtain \( A_{\phi,\varepsilon} \) as the reachable part. The latter is shown to be finite-state (Lem. \( \Sigma,\Omega \)).

**Definition 4.5 (the automata \( A_{\phi,\varepsilon} \))** Let \( \phi \) be an LTL-\( \Sigma \)-formula and \( \varepsilon \in (0,1) \). We define an alternating \([0,1]\)-acceptance automaton \( A_{\phi,\varepsilon} = (P(AP), Q, I, \delta, F) \) as follows. Its state space \( Q \) is \( \chi cl(\phi) \times [0,1]^+ \); hence a state is a pair \((\psi, d)\) of a formula and a discount sequence. The transition function \( \delta : Q \times P(AP) \rightarrow B^+(Q \cup [0,1]) \) is defined as follows. Let \( d = d_1d_2\ldots d_n \in [0,1]^+ \) and \( \sigma \in P(AP) \).

\[
\delta((\text{True}, d), \sigma) = d \boxdot 1.
\]

\[
\delta((p, d), \sigma) = \begin{cases} 
   d \boxdot 1 & \text{if } p \in \sigma, \\
   d \boxdot 0 & \text{otherwise}.
\end{cases}
\]

\[
\delta((\psi_1 \land \psi_2, d), \sigma) = \delta((\psi_1, d), \sigma) \land \delta((\psi_2, d), \sigma)
\]

\[
\delta((\psi_1 \lor \psi_2, d), \sigma) = \delta((\psi_1, d), \sigma) \lor \delta((\psi_2, d), \sigma)
\]

\[
\delta((\neg \psi, d), \sigma) = \delta((\neg \psi, d1), \sigma), \text{ where } d1 \text{ denotes the concatenation with } 1.
\]

\[
\delta((X\psi, d), \sigma) = (\psi, d).
\]

\[
\delta((\psi_1 \cup \psi_2, d), \sigma) = \begin{cases} 
   \delta((\psi_1, d), \sigma) \lor \delta((\psi_2, d), \sigma) & \text{if } |d| \text{ is odd,} \\
   \delta((\psi_1, d), \sigma) \land \delta((\psi_2, d), \sigma) & \text{otherwise}.
\end{cases}
\]

\[
\delta((\psi_1 \cup \psi_2, d), \sigma) = \begin{cases} 
   \delta((\psi_1 \cup \psi_2, d), \sigma) & \text{if } |d| \text{ is odd,} \\
   \delta((\psi_1 \cup \psi_2, d), \sigma) & \text{otherwise}.
\end{cases}
\]

\[
\delta((\psi_1 \cup \psi_2, d), \sigma) = \begin{cases} 
   \delta((\psi_1 \cup \psi_2, d), \sigma) & \text{if } |d| \text{ is odd,} \\
   \delta((\psi_1 \cup \psi_2, d), \sigma) & \text{otherwise}.
\end{cases}
\]

\[
\delta((\psi_1 \cup \psi_2, d), \sigma) = \begin{cases} 
   \delta((\psi_1 \cup \psi_2, d), \sigma) & \text{if } |d| \text{ is odd,} \\
   \delta((\psi_1 \cup \psi_2, d), \sigma) & \text{otherwise}.
\end{cases}
\]

For \( \delta((\psi_1 \cup_\eta \psi_2, d), \sigma) \) we make cases. Let \( d = d_1\ldots d_n \); if \( \eta(0) \cdot \prod_{i=1}^n d_i \leq \varepsilon \):

\[
\delta((\psi_1 \cup_\eta \psi_2, d), \sigma) = \begin{cases} 
   d \boxdot 0 & \text{if } |d| \text{ is odd,} \\
   d \boxdot \eta(0) & \text{otherwise}.
\end{cases}
\]
The set \( I \) of the initial states of \( A_{\varphi,\varepsilon} \) is \( \{ (\varphi, 1) \} \). The acceptance function \( F \) is
\[
F(\psi, \vec{d}) = \begin{cases} 
1 & \text{if } \psi = \psi_1 \cup \psi_2 \text{ and if } |\vec{d}| \text{ is even} \\
0 & \text{otherwise.}
\end{cases}
\]

(8)

The alternating \([0, 1]\)-acceptance automaton \( A_{\varphi,\varepsilon} \) is defined to be the restriction of \( A_{\varphi,\varepsilon}^p \) to the states that are reachable from the initial state \((\varphi, 1)\).

Examples of \( A_{\varphi,\varepsilon} \) are in Fig. 1-2 where \((\varphi, \varepsilon) = (G^{exp}_{exp\frac{1}{2}} F^{exp\frac{1}{3}} p, \frac{1}{3}) \) and \((F^{exp\frac{1}{2}} G^{exp\frac{1}{3}} p, \frac{1}{3}) \).

There a discount sequence \(d_1 \ldots d_n\) is denoted by \((d_1, \ldots, d_n)\) for readability.

Some remarks on Def. 4.5 are in order.

**In Absence of Discounting ( Sanity Check)**  If the formula \( \varphi \) contains no discounting operator \( U \eta \), then the construction essentially coincides the usual one in [18] that translates a (usual) LTL formula to an alternating Büchi automaton. To see it, note that the parity of the length \(|\vec{d}|\) of a discount sequence indicates whether we are looking at the
formula itself or actually its negation (cf. the definition of \( \delta((\neg \psi, \vec{d}), \sigma) \)). Therefore, in the first case of (8), \( |\vec{d}| \) being even means that we are in fact dealing with a greatest fixed point. This makes the state accepting, much like in [18].

**A\(\varphi, \varepsilon\) is Quantitative** The acceptance values of the states of \( A_{\varphi, \varepsilon} \) are Boolean (see (8)). Nevertheless the automaton is quantitative, in that non-Boolean values from \([0, 1]\) appear as atomic propositions in the range \( B^+(Q \cup [0, 1]) \) of the transition \( \delta \) (they occur at the leaves in Fig. 1–2). Once we transform \( A_{\varphi, \varepsilon} \) to a non-alternating automaton (Lem. 3.4), these non-Boolean values give rise to non-Boolean acceptance values.

**Event Horizon** A fundamental idea from [2]—it plays a crucial role in our translation too—is that a discounting operator, in presence of a threshold (in [2]) or a nonzero margin (here), allows an exact representation by a (finitary) formula without a fixed point operator. The latter means, for example:

\[
\begin{align*}
[\pi, F^{\exp_2}_{\varphi}] \geq \frac{1}{4} & \iff \pi \models \varphi \lor X\varphi \lor XX\varphi, \quad \text{and} \quad (9) \\
[\pi, G^{\exp_2}_{\varphi}] \geq \frac{3}{4} & \iff \pi \models \varphi \land X\varphi \land XX\varphi, \quad (10)
\end{align*}
\]

and so on. Note that in (9), whatever happens after two time units has contributions less than \( (\frac{1}{2})^2 = \frac{1}{4} \) and therefore never enough to make up the threshold. The example (10) is similar, with events in the future having only negligible negative contributions. In other words: fixed point operators with discounting have an **event horizon**—in the above examples (9) and (10) it lies between \( t = 2 \) and 3—nothing beyond which matters.
This idea of event horizon is used in the distinction between (6) and (7). The value 
\( \eta(0) \cdot \prod_{i=1}^{n} d_i \) is, as we shall see, the greatest contribution to a truth value that the events henceforth potentially have. In case it is smaller than the margin \( \varepsilon \) we can safely ignore the positive contribution henceforth and take the smallest possible truth value \( 0 \)—much like the disjunct \( X^3 \phi \lor X^4 \phi \lor \cdots \) is truncated in (9). This is what is done in the first case in (6). The second case in (6) is about a greatest fixed point and we truncate the negative contributions of the events beyond the event horizon—this is much like the obligation \( X^3 \phi \land X^4 \phi \land \cdots \) is lifted in (10). In this case we use the greatest truth value possible, namely \( \eta(0) \). This is what is done in (6).

**Use of Discount Sequences**

Discount sequences \( \vec{d} \) are used for two purposes. Firstly, as we already described, its length \( |\vec{d}| \) indicates the alternation between positive and negative views on a formula. Consequently many clauses in the definition of \( \delta \) distinguish cases according to the parity of \( |\vec{d}| \). Secondly it records all the discount factors that have been encountered. See (7), where the last element of \( \vec{d} \) is multiplied by the newly encountered factor \( \eta(0) \) and updated to \( \vec{d} \odot \eta(0) \).

Such accumulation \( \vec{d} \) of discount factors acts on a truth value using the \( \boxdot \) operator, like in (6) and in the definition of \( \delta((\text{True}, \vec{d}), \sigma) \). Note that discount factors work differently depending on polarity: for \( U \eta \) formulas they discount positive contributions; for their negation they discount negative contributions. See the definition (3) of \( \boxdot \).

**Lemma 4.6** The automaton \( A_{\phi, \varepsilon} \) has only finitely many states. \( \square \)

The following “correctness lemma” claims that \( A_{\phi, \varepsilon} \) conducts the expected task. See Appendix A.3 for its (extensive) proof.

**Lemma 4.7** Let \( \phi \) be an LTL\(_{\text{disc}}[D] \) formula and \( \varepsilon \in (0, 1) \) be a positive real number. For each computation \( \pi \in (\mathcal{P}(AP))^\omega \), we have
\[
[\pi, \phi] - \varepsilon \leq \mathcal{L}(A_{\phi, \varepsilon})(\pi) \leq [\pi, \phi].
\]

We can translate \( A_{\phi, \varepsilon} \) to a (non-alternating) \( [0, 1] \)-acceptance automaton (Lem. 3.4).

**Corollary 4.8** Let \( \phi \) be an LTL\(_{\text{disc}}[D] \) formula and \( \varepsilon \in (0, 1) \) be a positive real number. There exists a (non-alternating) \( [0, 1] \)-acceptance automaton \( A_{\phi, \varepsilon}^\text{na} \) such that
\[
[\pi, \phi] - \varepsilon \leq \mathcal{L}(A_{\phi, \varepsilon}^\text{na})(\pi) \leq [\pi, \phi]
\]
for each computation \( \pi \in (\mathcal{P}(AP))^\omega \). \( \square \)

Towards the solution of the near-optimal path synthesis problem (Def. 4.2), we construct the product of \( A_{\phi, \varepsilon}^\text{na} \) in Cor. 4.8 and the given Kripke structure \( K \). Since transitions of \( [0, 1] \)-acceptance automata are nondeterministic, this product can be defined similarly to the product of a (Boolean) automaton and a Kripke structure.
**Definition 4.9** Let $A = (\mathcal{P}(AP), Q, I, \delta, F)$ be a $[0,1]$-acceptance automaton and $K = (W, R, \lambda)$ be a Kripke structure. Their product $A \times K$ is a $[0,1]$-acceptance automaton $(1, Q', I', \delta', F')$—over a singleton alphabet $1 = \{\bullet\}$—defined by

\[
Q' = Q \times W , \quad I' = I \times W , \quad \delta'( (q, s), \bullet ) = \{ (q', s') \mid q' \in \delta(q, \lambda(s)), (s, s') \in R \} , \quad F'(q, s) = F(q) .
\]

**Lemma 4.10** Let $(q_0, s_0) \bullet (q_1, s_1) \bullet \ldots$ be an optimal run of the automaton $A \times K$ (that necessarily exists by Lem. 3.2). The path $s_0 s_1 \ldots \in \text{path}(K)$ realizes the optimal value of $A$, that is,

\[
L(A)(\lambda(s_0) \lambda(s_1) \ldots) = \max_{\xi \in \text{path}(K)} L(A)(\lambda(\xi)) .
\]

Finally we present an algorithm for the near-optimal path synthesis problem.

**Theorem 4.11 (main theorem)** In the setting of Def. 4.2, let $(q_0, s_0) \bullet (q_1, s_1) \bullet \ldots$ be an optimal run (computed by Lem. 3.2) for the $[0,1]$-acceptance automaton $A_{\varphi, \varepsilon} \times K$ constructed as in Def. 4.5, Cor. 4.8 and Def. 4.9. Then the path $s_0 s_1 \ldots \in \text{path}(K)$ is a solution to the near-optimal path synthesis problem (Def. 4.2).

Moreover, the solution $s_0 s_1 \ldots$ can be chosen to be ultimately periodic. \hfill \square

We can synthesize a near-worst path, too, by seeking for an optimal path for $\neg \varphi$.

**5 Complexity**

The size of the automata $A_{\varphi, \varepsilon}$ constructed in Cor. 4.8 varies depending on the discounting functions used in $\varphi$. Here we restrict to exponential discounting functions (Def. 2.1), and to rational $\lambda$. This is done in the complexity analysis in [2], too.

Recall that $E = \{ \exp{\lambda} \mid \lambda \in (0, 1) \cap \mathbb{Q} \}$ is the set of such discounting functions (Def. 2.1). We use the notion $|\langle \varphi \rangle|$ of the size of a formula $\varphi$ introduced in [2]: it reflects the description length of $\lambda \in \mathbb{Q}$ that appears in the discounting functions in $\varphi$, as well as the length of the formula $\varphi$ as an expression.

**Proposition 5.1** Let $\varphi$ be an LTL\textsuperscript{disc}[$E$] formula and $\varepsilon \in (0, 1) \cap \mathbb{Q}$ be a positive rational number. The size of the state space of the alternating $[0,1]$-acceptance automaton $A_{\varphi, \varepsilon}$ is singly exponential in $|\langle \varphi \rangle|$ and in the length of the description of $\varepsilon$. \hfill \square

In the procedure in Thm. 4.11 we apply Lem. 3.4 to turn $A_{\varphi, \varepsilon}$ into a non-alternating automaton $A_{\varphi, \varepsilon}^{\text{na}}$. This however incurs an exponential blowup, making the size of $A_{\varphi, \varepsilon}^{\text{na}}$ doubly exponential in $|\langle \varphi \rangle|$. In fact further optimization is possible in the construction of $A_{\varphi, \varepsilon}^{\text{na}}$—by careful inspection of the structure of $A_{\varphi, \varepsilon}$—and a blowup is avoided.

**Theorem 5.2** The near-optimal path synthesis problem for LTL\textsuperscript{disc}[$E$] is: in PSPACE in $|\langle \varphi \rangle|$ and in the description length of $\varepsilon$; and in NLOGSPACE in the size of $K$. \hfill \square
6 Conclusions and Future Work

For the quantitative logic LTL^{disc}_{D} with future discounting [2], we formulated a natural problem of synthesizing near-optimal schedulers, and presented an algorithm. The latter relies on: the existing idea of event horizon exploited in [2] for the threshold model checking problem, as well as a supposedly widely-applicable technique of translation to [0, 1]-acceptance automata and a lasso-style optimal value algorithm for them.

Here are several directions of future work.

**Controller Synthesis for Open Systems**

We note that the current results are focused on closed systems. For open or reactive systems (like a server that responds to requests that come from the environment) we would wish to synthesize a controller—formally a strategy or a transducer—that achieves a near-optimal performance.

An envisaged workflow, following the one in [18], is as follows. We will use the same automaton $A_{ϕ,ε}$ (Def. 4.5). It is then: 1) determinized, 2) transformed into a tree automaton that accepts the desired strategies, and 3) the optimal value of the tree automaton is checked, much like in Lem. 3.2. While the step 2) will be straightforward, the steps 1) and 3) (namely: determinization of [0, 1]-acceptance automata, and the optimal value problem for “[0, 1]-acceptance Rabin automata”) are yet to be investigated. Another possible workflow is by an adaptation of the Safraless algorithm [13].

**Probabilistic Systems and LTL^{disc}_{D}**

Here and in [2] the system model is a Kripke structure that is nondeterministic. Adding probabilistic branching will gives us a set of new problems to be solved: for Markov chains the threshold model-checking problem can be formulated; for Markov decision processes, we have both the threshold model-checking problem and the near-optimal scheduler synthesis problem. Furthermore, another axis of variation is given by whether we consider the expected value or the worst-case value. In the latter case we would wish to exclude truth values that arise with probability 0. All these variations have important applications in various areas.

**Comparison against Binary Search by Threshold Model-Checking**

There is in fact a straightforward algorithm for the near-optimal scheduler synthesis problem studied in this paper. It conducts a binary search for the near-optimal truth value, by repeating the threshold model-checking algorithm in [2], for thresholds: $\frac{1}{2}$, $\frac{1}{4}$ or $\frac{3}{8}$, $\frac{1}{8}$, ..., $\frac{6}{8}$ or $\frac{7}{8}$, and so on. Given a margin $ε \in (0, 1)$, we need $-\log ε$ rounds.

The performance comparison between this binary search method and our algorithm will be best done by experiments, which we leave as future work. We nevertheless believe that our use of [0, 1]-acceptance automata and the optimal value algorithm (Lem. 3.2)—a technique we believe to be fundamental for this kind of tasks—give us a substantial advantage.

Acknowledgments Thanks are due to Shuichi Hirahara for useful discussions. The authors are supported by Grants-in-Aid for Young Scientists (A) No. 24680001, JSPS.

References

1. Shaull Almagor, Udi Boker, and Orna Kupferman. Formalizing and reasoning about quality. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga,
1. Latvian July 8-12, 2013, Proceedings, Part II, volume 7966 of Lecture Notes in Computer Science, pages 15–27. Springer, 2013.

2. Shaull Almagor, Udi Boker, and Orna Kupferman. Discounting in LTL. In Erika Ábrahám and Klaus Havelund, editors, Tools and Algorithms for the Construction and Analysis of Systems - 20th International Conference, TACAS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings, volume 8413 of Lecture Notes in Computer Science, pages 424–439. Springer, 2014.

3. Christel Baier, Clemens Dubslaff, and Sascha Klüppelholz. Trade-off analysis meets probabilistic model checking. In Thomas A. Henzinger and Dale Miller, editors, Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS ’14, Vienna, Austria, July 14 - 18, 2014, page 1. ACM, 2014.

4. Roderick Bloem, Krishnendu Chatterjee, Thomas A. Henzinger, and Barbara Jobstmann. Better quality in synthesis through quantitative objectives. In Ahmed Bouajjani and Oded Maler, editors, Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings, volume 5643 of Lecture Notes in Computer Science, pages 140–156. Springer, 2009.

5. Patricia Bouyer, Nicolas Markey, and Raj Mohan Matteplackel. Averaging in LTL. In Paolo Baldan and Daniele Gorla, editors, CONCUR 2014 - Concurrency Theory - 25th International Conference, CONCUR 2014, Rome, Italy, September 2-5, 2014. Proceedings, volume 8704 of Lecture Notes in Computer Science, pages 266–280. Springer, 2014.

6. Pavol Cerný, Krishnendu Chatterjee, Thomas A. Henzinger, Arjun Radhakrishna, and Roshit Singh. Quantitative synthesis for concurrent programs. In Ganesh Gopalakrishnan and Shaz Qadeer, editors, Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings, volume 6806 of Lecture Notes in Computer Science, pages 243–259. Springer, 2011.

7. Krishnendu Chatterjee, Thomas A. Henzinger, and Marcin Jurdzinski. Mean-payoff parity games. In 20th IEEE Symposium on Logic in Computer Science (LICS 2005), 26-29 June 2005, Chicago, IL, USA, Proceedings, pages 178–187. IEEE Computer Society, 2005.

8. Ling Cheung, Marielle Stoelinga, and Frits W. Vaandrager. A testing scenario for probabilistic processes. J. ACM, 54(6), 2007.

9. Luca de Alfaro, Thomas A. Henzinger, and Rupak Majumdar. Discounting the future in systems theory. In Jos C. M. Baeten, Jan Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger, editors, Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, Eindhoven, The Netherlands, June 30 - July 4, 2003. Proceedings, volume 2719 of Lecture Notes in Computer Science, pages 1022–1037. Springer, 2003.

10. Manfred Droste and Ulrike Püschmann. On weighted Büchi automata with order-complete weights. JAC, 17(2):235–260, 2007.

11. Marco Faella, Axel Legay, and Mariëlle Stoeingla. Model checking quantitative linear time logic. Electr. Notes Theor. Comput. Sci., 220(3):61–77, 2008.

12. Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. Formal Asp. Comput., 6(5):512–535, 1994.

13. Orna Kupferman, Nir Piterman, and Moshe Y. Vardi. Saferless compositional synthesis. In Thomas Ball and Robert B. Jones, editors, Computer Aided Verification, 18th International Conference, CAV 2006, Seattle, WA, USA, August 17-20, 2006, Proceedings, volume 4144 of Lecture Notes in Computer Science, pages 31–44. Springer, 2006.

14. Satoru Miyano and Takeshi Hayashi. Alternating finite automata on omega-words. Theor. Comput. Sci., 32:321–330, 1984.
15. Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In Conference Record of the Sixteenth Annual ACM Symposium on Principles of Programming Languages, Austin, Texas, USA, January 11-13, 1989, pages 179–190. ACM Press, 1989.

16. George Rahonis. Infinite fuzzy computations. Fuzzy Sets and Systems, 153(2):275–288, 2005.

17. R. J. van Glabbeek. The linear time–branching time spectrum I; the semantics of concrete, sequential processes. In J. A. Bergstra, A. Ponse, and S. A. Smolka, editors, Handbook of Process Algebra, chapter 1, pages 3–99. Elsevier, 2001.

18. Moshe Y. Vardi. An automata-theoretic approach to linear temporal logic. In Logics for Concurrency: Structure versus Automata, volume 1043 of Lecture Notes in Computer Science, pages 238–266. Springer-Verlag, 1996.
A Omitted Proofs

A.1 Proof of Lem. 3.4, and Further Optimization

Proof. We first describe the formal construction; intuitions follow shortly.

Without loss of generality, we can assume that a positive Boolean formula \( \delta(q, a) \) is a disjunctive normal form; therefore the transition function is of the type \( \delta : Q \times \Sigma \to \mathcal{P}(\mathcal{P}(Q \cup \{0, 1\})) \). More concretely, for each \( q \in Q \) and \( a \in \Sigma \), the formula \( \delta(q, a) \) is a disjunction of formulas of the form

\[
(q_1 \land \cdots \land q_k) \land (v_1 \land \cdots \land v_l)
\]

where \( q_j \in Q \) and \( v_j \in \{0, 1\} \) are atomic propositions (we changed their order suitably). Moreover, since the conjunction \( v_1 \land \cdots \land v_l \) is equivalent to a single atomic proposition \( \min\{v_1, \ldots, v_l\} \), we assume that any disjunct of the DNF formula \( \delta(q, a) \) is of the form

\[
(q_1 \land \cdots \land q_k) \land v.
\]

Let \( V_Q = \{ F(q) \mid q \in Q \} \) be the set of acceptance values that occur in \( A \), and \( V_\delta \) be the set of values from \( [0, 1] \) (i.e. atomic propositions from \( [0, 1] \)) that occur in the transition function \( \delta \), that is,

\[
V_\delta = \bigcup_{q \in Q, a \in \Sigma} \{ v \mid ((q_1 \land \cdots \land q_k) \land v) \in \delta(q, a) \}.
\]

We define \( A' = (\Sigma', Q', I', \delta', F') \) as follows.

\[
Q' = \mathcal{P}(Q \times V_Q) \times V_1 \times \{ \mathbf{ff}, \mathbf{tt} \},
\]

\[
I' = \{ \{(q_0, F(q_0)), 1, \mathbf{ff}\} \mid q_0 \in I \},
\]

\[
F'(Y, v, b) = \begin{cases}
\min\{v, \min\{v' \mid \exists q \in Q. (q, v') \in Y\} \} & \text{if } b = \mathbf{tt} \\
0 & \text{otherwise}.
\end{cases}
\]

The transition function \( \delta' \) is defined as follows. Let \( \vec{q} = \{ (q_1^1, v_1^1), \ldots, (q_n^a, v_n^a) \}, v, b \) be a state in \( Q' \), and \( a \in \Sigma \). Then \( \delta'(\vec{q}, a) \) is defined, in case \( b = \mathbf{ff} \), by:

\[
\left\{ \begin{array}{c}
\left( \begin{array}{c}
(q_1^1, \max\{v^1, F(q_1^1)\}), \ldots, (q_k^1, \max\{v^1, F(q_k^1)\})
\end{array} \right), \\
\vdots \\
\left( \begin{array}{c}
(q_1^n, \max\{v^n, F(q_1^n)\}), \ldots, (q_k^n, \max\{v^n, F(q_k^n)\})
\end{array} \right), \\
\min\{v, u_1^1, \ldots, u_n^a\},
\end{array} \right.
\]

\[
\left| (q_1^i \land \cdots \land q_k^i) \land u_i^i \in \delta(q^i, a), \ b' \in \{\mathbf{tt}, \mathbf{ff}\} \right.
\]

(11)
in case \( b = \text{tt} \),
\[
\begin{pmatrix}
\{ (q_1^1, F(q_1^1)), \ldots, (q_1^n, F(q_1^n)) \} \\
The set \( \{ q^1, v^1 \}, \ldots, (q^n, v^n) \} \) stands for the conjunction of these pairs.
- The second component \( v \in [0, 1] \) of \( \bar{q} \) is for keeping track of: the values at the leaves of the corresponding run tree, more precisely the smallest among such.
- The flag \( b \in \{ \text{ff}, \text{tt} \} \) is called an exposition flag; it determines if the internally accumulated acceptance values \( v^1, \ldots, v^n \) should be exposed or not. Note the definition of \( F' \): the acceptance value of a state of \( \mathcal{A}' \) is nonzero only if the exposition flag \( b \) is \( \text{tt} \).

Let us comment on the definition of the transition function \( \delta' \). Starting from \( \bar{q} = \{(q^1, v^1), \ldots, (q^n, v^n)\}, v, b\)—in which the “current state” is the conjunction \( q^1 \land q^2 \land \cdots \land q^n \)—we choose one disjunct \( q^1 \land \cdots \land q_i \in \delta(q^i, a) \) for each \( q^i \) and the “next state” is
\[
(q_1^1 \land \cdots \land q_1^n) \land (q_2^1 \land \cdots \land q_2^n) \land \cdots \land (q_i^1 \land \cdots \land q_i^n).
\]

If the exposition flag \( b \) is \( \text{ff} \) then we keep accumulating the acceptance values that we have seen since the last exposition, resulting in the occurrence of max in (11). If the flag is \( \text{tt} \) then the internally accumulated acceptance values are “used” (see the definition of \( F' \)), and these values must be “forgotten” so that we simulate a Büchi-like acceptance condition for \( \mathcal{A} \). Therefore in (12), there are no \( v^1, \ldots, v^n \) occurring and we have a fresh start.

The state space \( Q' \) of \( \mathcal{A}' \) in the previous proof can actually be smaller: we can identify two states \((Y, v, b)\) and \((Y', v, b)\) if \( \min\{v' \in V_Q \mid (q, v') \in Y\} = \min\{v' \in V_Q \mid (q, v') \in Y'\} \).
\( V_Q \mid (q, v') \in Y' \) holds for each \( q \in Q \)—this is the case for example when \( Y = \{(q, \frac{2}{3}), (q, 1)\} \) and \( Y' = \{(q, \frac{2}{3})\} \). Therefore we only need states \((Y, v, b)\) such that \( \forall (q, v), (q', v') \in Y. (q = q' \Rightarrow v = v') \), that is, \( Y \) can be regarded as a partial function. Summarizing, we can reduce the state space to \((V_Q \cup \{\ast\})^Q \times V_g \times \{\ast, \ast\}\). The size of the first component is \( 2^{|Q| \times \log |V_Q|} \), while it was \( 2^{|Q| \times |V_Q|} \) before this optimization.

A.2 Proof of Lem. 4.6

**Proof.** The state space \( Q = \text{xcl}(\varphi) \times [0, 1]^+ \) of \( A_{\varphi, \varepsilon}^p \) is infinite for three reasons: 1) the extended closure \( \text{xcl}(\varphi) \) contains \( \varphi_1 \cup \varphi_2 \) for unbounded \( k \in \mathbb{N} \) (see (2)); 2) discount factors occurring in \( d \in [0, 1]^+ \) are multiples of numbers from an infinite set \( \{\eta(0), \eta(1), \ldots\} \); and 3) the length of a discount sequence \( d \in [0, 1]^+ \) is potentially unbounded.

We can easily see that the reason 3) is not a problem for us: in the construction of \( A_{\varphi, \varepsilon}^p \) (Def. 4.5), the length of a discount sequence \( d \) grows only when we encounter negation (i.e. in the definition of \( \delta((-\psi, d), \sigma) \)). Therefore in a reachable state \((\psi, d)\) of \( A_{\varphi, \varepsilon}^p \), the length of \( d \) is bounded by the number of negation operators occurring in \( \varphi \).

To see that the reasons 1) and 2) are non-problematic either, note that we obtain new states for these reasons only in the clause (7) of the definition of \( \delta((\psi_1 \cup \psi_2, d), \sigma) \). This clause is applied only when \((\eta(0) \cdot \prod_{i=1}^{k} d_i > \varepsilon) \), a condition satisfied by only finitely many reachable states of \( A_{\varphi}^p \):

- The discount function \( \eta \) here is of the form \( \eta = (\eta')^{+k} \), where \( \eta' \) occurs in the original formula \( \varphi \) and \( k \in \mathbb{N} \). Since a discounting function \( \eta' \) tends to 0 (Def. 2.1), \( \eta(0) = (\eta')^{+k}(0) = \eta'(k) \) tends to 0 as \( k \to \infty \), too, making only finitely many \( k \) suitable.
- Each discount factor \( d_j \) in \( d \) is a multiple \( \eta_1(k_1) \times \cdots \times \eta_m(k_m) \), where \( \eta_i \) is a discounting function occurring in \( \varphi \) and \( k_i \in \mathbb{N} \). They must at least satisfy \( \eta_i(k_i) > \varepsilon \): since \( \eta_i \) tends to 0, this allows only finitely many choices of \( k_i \), for each \( \eta_i \). Furthermore, the (necessary) condition that \( d_j = \eta_1(k_1) \times \cdots \times \eta_m(k_m) > \varepsilon \) bounds the length \( m \) of the multiple, too. \( \square \)

A.3 Proof of Lem. 4.7

**Proof.** In what follows let \( Q \) denote the state space of \( A_{\varphi, \varepsilon} \); \( \delta \) denote its transition function; and \( F \) denote its acceptance function. For each \((\psi, d) \in Q\), we define an alternation \([0, 1]\)-acceptance automaton \( A_{\psi, \varepsilon}^{(\psi, d)} \) by changing the initial state to \((\psi, \bar{d})\), that is, \( A_{\psi, \varepsilon}^{(\psi, d)} = (\mathcal{P}(AP), Q, \{(\psi, d)\}, \delta, F) \). Suppose that \( \bar{d} = d_1 d_2 \ldots d_n \). We prove the following more general statement, inductively on the construction of \( \psi \):

\[ \bar{d} \boxtimes [\pi, \psi] - \varepsilon \leq L(A_{\psi, \varepsilon}^{(\psi, d)})(\pi) \leq \bar{d} \boxtimes [\pi, \psi] \tag{13} \]

for each \( \pi \in (\mathcal{P}(AP))^* \).

The cases where \( \psi = \text{True}, p, p_1 \wedge p_2, \neg \psi' \) or \( X \psi' \) are straightforward. Here we only prove the case where \( \psi = \neg \psi' \). By the definition of the automaton \( A_{\varphi, \varepsilon} \)
we have $\mathcal{L}(A_{\overset{\circ}{\varphi},\varepsilon}^{\psi,d})(\pi) = \mathcal{L}(A_{\overset{\circ}{\varphi},\varepsilon}^d)(\pi)$, and the latter value lies in the interval $[\overline{(d\overline{1}) \boxtimes [\pi,\psi']} - \varepsilon, (d\overline{1}) \boxtimes [\pi,\psi']]$ by the induction hypothesis. Now we obtain
\[
(d\overline{1}) \boxtimes [\pi,\psi'] = \bar{d} \boxtimes (1 - [\pi,\psi']) = \bar{d} \boxtimes [\pi,\neg\psi'],
\]
as required. Here the former equality is due to the definition of $\boxtimes$; the latter is the semantics of $\neg\psi$.

Suppose $\psi = \psi_1 U \psi_2$; we first deal with the case when $|\bar{d}|$ is odd. Let $\pi \in (\mathcal{P}(AP))^\omega$. We note that, since $|\bar{d}|$ is odd, the function $\bar{d} \boxtimes (\underline{\underline{\bot}}) : [0,1] \to [0,1]$ is monotone and continuous (see (4)). This is used in:
\[
\bar{d} \boxtimes [\pi,\psi_1 U \psi_2] = \bar{d} \boxtimes \sup_{i \in \mathbb{N}} \min \{ [\pi^i,\psi_1], \min_{0 \leq j \leq i-1} [\pi^j,\psi_1] \} \\
= \sup_{i \in \mathbb{N}} \min \{ \bar{d} \boxtimes [\pi^i,\psi_2], \min_{0 \leq j \leq i-1} (\bar{d} \boxtimes [\pi^j,\psi_1]) \}.
\]

Now let us take a closer look at how the value $\mathcal{L}(A_{\overset{\circ}{\varphi},\varepsilon}^{\psi_1 U \psi_2,d})(\pi)$ is defined for an alternating $[0,1]$-acceptance automaton $A_{\overset{\circ}{\varphi},\varepsilon}^{\psi_1 U \psi_2,d}$. As seen in Def. 3.3, the notions of run tree and path are Boolean; a non-Boolean value arises for the first time as the “utility” $F^\infty(\rho)$ of a path $\rho$ of a run tree. According to Def. 4.5 of $A_{\overset{\circ}{\varphi},\varepsilon}$ (in particular, the definition of $\delta((\psi_1 U \psi_2,d),\pi)$), any possible run tree $\tau$ from the state $(\psi_1 U \psi_2,d)$ is of one of the following forms:

- the second disjunct $\delta((\psi_1,d),\pi)$ is chosen all the way (Fig. 3 left),
- or
- the first disjunct $\delta((\psi_2,d),\pi)$ is eventually hit (Fig. 3 right).

In the former case, the utility $\min_{\rho \in \text{path}(\tau)} F^\infty(\rho)$ of such a run tree $\tau$ is given by
\[
\min_{\rho \in \text{path}(\tau)} F^\infty(\rho) = \inf_{j \in \mathbb{N}} \mathcal{L}(A_{\overset{\circ}{\varphi},\varepsilon}^{\psi_j,d})(\pi^j),
\]
where the first value $F((\psi_1 U \psi_2,d)$ is induced by the rightmost path in Fig. 3 left. We have $F((\psi_1 U \psi_2,d) = 0$ by definition (see (8)); therefore the utility obtained in this case is 0.

In the latter case, assume that the second disjunct $\delta((\psi_2,d),\pi)$ is hit at depth $i$. The tree’s utility is then given by
\[
\min_{\rho \in \text{path}(\tau)} F^\infty(\rho) = \min_{0 \leq j \leq i-1} \mathcal{L}(A_{\overset{\circ}{\varphi},\varepsilon}^{\psi_j,d})(\pi^j),
\]
where, again, the first value $\mathcal{L}(A_{\overset{\circ}{\varphi},\varepsilon}^{\psi_1,d})(\pi^i)$ arises from the rightmost path in Fig. 3 right.

Putting all these together, we have
\[
\mathcal{L}(A_{\overset{\circ}{\varphi},\varepsilon}^{\psi_1 U \psi_2,d})(\pi) = \sup_{i \in \mathbb{N}} \left( \min_{\rho \in \text{path}(\tau)} F^\infty(\rho) \right) = \mathcal{L}(A_{\overset{\circ}{\varphi},\varepsilon}^{\psi_1 U \psi_2,d})(\pi^i)
\]
by the induction hypothesis
\[
= \left[ \bar{d} \boxtimes [\pi,\psi_1 U \psi_2] - \varepsilon, \bar{d} \boxtimes [\pi,\psi_1 U \psi_2] \right].
\]
as required.

Suppose that \( \psi = \psi_1 U \psi_2 \) and that \( |\vec{d}| \) is even. Let \( \pi \in (\mathcal{P}(AP))^\omega \). Since \( \vec{d} \otimes (\_\_\_) \) is antitone and continuous, the second equality below holds.

\[
\vec{d} \otimes [\pi, \psi_1 U \psi_2] = \inf_{i \in \mathbb{N}} \left( \max \{ \vec{d} \otimes [\pi, \psi_2], \min_{0 \leq j \leq i-1} [\pi^j, \psi_1] \} \right) .
\]

(15)

We use the following observation. It is a quantitative adaptation of the classic duality between the temporal operators \( U \) and \( R \) (“release”).

**Sublemma A.1** Let \( a_0, a_1, \ldots \) and \( b_0, b_1, \ldots \) all be real numbers in \([0,1]\). We have

\[
\inf_{i \in \mathbb{N}} \left( \max \{ b_i, \max_{0 \leq j \leq i-1} a_j \} \right) = \left\{ \sup_{j \in \mathbb{N}} \left( \min \{ a_j, \min_{0 \leq i \leq j} b_i \} \right) , \inf_{i \in \mathbb{N}} b_i \right\} ,
\]

that is, denoting binary \( \min \) and \( \max \) by \( \land \) and \( \lor \):

\[
\inf_{i \in \mathbb{N}} \left( b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1}) \right) = \left( \sup_{j \in \mathbb{N}} \left( a_j \land (b_0 \land b_1 \land \cdots \land b_j) \right) \right) \lor \inf_{i \in \mathbb{N}} b_i .
\]

(16)

**Proof.** (Of Sublem. A.1) We distinguish two cases. Let us first assume that there exists \( i \in \mathbb{N} \) such that \( b_i < a_0 \lor a_1 \lor \cdots \lor a_{i-1} \). Let \( k \) be the least number among such, that is, \( k \) satisfies that

\[
b_k < a_0 \lor a_1 \lor \cdots \lor a_{k-1} \quad \text{and} \quad \forall i \in [0, k-1], b_i \geq a_0 \lor a_1 \lor \cdots \lor a_{i-1} .
\]

(17)
Moreover, let $l \in [0, k - 1]$ be a number such that $a_l = a_0 \lor a_1 \lor \cdots \lor a_{k-1}$. We have
\[
\inf_{i \in \mathbb{N}} \big( b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1}) \big)
= b_0 \land (b_1 \lor a_0) \land (b_2 \lor a_0 \lor a_1) \land \cdots \\
\land (b_k \lor a_0 \lor \cdots \lor a_{k-1}) \land \inf_{i \geq k+1} \big( b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1}) \big)
= b_0 \land b_1 \land \cdots \land b_{k-1} \land a_l \land \inf_{i \geq k+1} \big( b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1}) \big)
\]
by def. of $k$. \(17\).

Since we have $a_l \leq b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1})$ for each $i \in [k+1, \infty)$,
\[
a_l \leq \inf_{i \geq k+1} \big( b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1}) \big)
\]
and we obtain
\[
\inf_{i \in \mathbb{N}} \big( b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1}) \big) = b_0 \land b_1 \land \cdots \land b_{k-1} \land a_l . \quad (18)
\]

Now we compare the last value $b_0 \land b_1 \land \cdots \land b_{k-1} \land a_l$ with the right-hand side of our goal \(16\). By the definition of $k$ and $l$, for each $j \in [0, k - 1]$, we have
\[
a_j \leq a_0 \lor a_1 \lor \cdots \lor a_{k-1} = a_l \quad \text{and} \quad \forall i \in [j+1, k-1], \ a_j \leq a_0 \lor a_1 \lor \cdots \lor a_{i-1} \leq b_i ,
\]
yielding
\[
a_j \leq a_l \land b_{j+1} \land b_{j+2} \land \cdots \land b_{k-1} , \quad \text{and hence}
\]
\[
a_j \land (b_0 \land b_1 \land \cdots \land b_j) \leq b_0 \land b_1 \land \cdots \land b_j \land b_{j+1} \land \cdots \land b_{k-1} \land a_l .
\]
The last inequality holds for each $j \in [k, \infty)$, too:
\[
a_j \land (b_0 \land b_1 \land \cdots \land b_j) \leq b_0 \land b_1 \land \cdots \land b_{k-1} \land b_k \leq b_0 \land b_1 \land \cdots \land b_{k-1} \land (a_0 \lor a_1 \lor \cdots \lor a_{k-1})
\]
by def. of $k$, \(17\)
\[
= b_0 \land b_1 \land \cdots \land b_{k-1} \land a_l \quad \text{by def. of } l .
\]
Consequently
\[
\sup_{j \in \mathbb{N}} \big( a_j \land (b_0 \land b_1 \land \cdots \land b_j) \big) \leq b_0 \land b_1 \land \cdots \land b_{k-1} \land a_l . \quad (19)
\]

We turn to the other part $\inf_{i \in \mathbb{N}} b_i$ of the right-hand side of \(16\). By the definition of $k$ and $l$, we have $b_k \leq a_0 \lor a_1 \lor \cdots \lor a_{k-1} = a_l$. Therefore
\[
\inf_{i \in \mathbb{N}} b_i \leq b_0 \land b_1 \land \cdots \land b_{k-1} \land b_k \leq b_0 \land b_1 \land \cdots \land b_{k-1} \land a_l . \quad (20)
\]

By \(19\) and \(20\),
\[
\left( \sup_{j \in \mathbb{N}} \big( a_j \land (b_0 \land b_1 \land \cdots \land b_j) \big) \right) \lor \inf_{i \in \mathbb{N}} b_i \leq b_0 \land b_1 \land \cdots \land b_{k-1} \land a_l , \quad (21)
\]
on the one hand. On the other hand, since \( l \in [0, k - 1] \),
\[
\left( \sup_{j \in \mathbb{N}} (a_j \land (b_0 \land b_1 \land \cdots \land b_j)) \right) \lor \inf_{i \in \mathbb{N}} b_i \geq \sup_{j \in \mathbb{N}} (a_j \land (b_0 \land b_1 \land \cdots \land b_j)) \\
\geq a_l \land (b_0 \land b_1 \land \cdots \land b_l) \\
\geq a_l \land (b_0 \land b_1 \land \cdots \land b_{k-1}) .
\]
\[ (22) \]

By (21) and (22),
\[
\left( \sup_{j \in \mathbb{N}} (a_j \land (b_0 \land b_1 \land \cdots \land b_j)) \right) \lor \inf_{i \in \mathbb{N}} b_i = b_0 \land b_1 \land \cdots \land b_{k-1} \land a_l .
\]
\[ (23) \]

By (18) and (23),
\[
\inf_{i \in \mathbb{N}} (b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1})) = \left( \sup_{j \in \mathbb{N}} (a_j \land (b_0 \land b_1 \land \cdots \land b_j)) \right) \lor \inf_{i \in \mathbb{N}} b_i .
\]
This establish the claim, in our first case where there exists \( i \in \mathbb{N} \) such that \( b_i < a_0 \lor a_1 \lor \cdots \lor a_{i-1} \).

In the other case we assume that \( b_i \geq a_0 \lor a_1 \lor \cdots \lor a_{i-1} \) for each \( i \in \mathbb{N} \). By this assumption, \( b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1}) = b_i \) for each \( i \in \mathbb{N} \). Therefore
\[
\inf_{i \in \mathbb{N}} (b_i \lor (a_0 \lor a_1 \lor \cdots \lor a_{i-1})) = \inf_{i \in \mathbb{N}} b_i .
\]
\[ (24) \]

Let us now fix \( j \in \mathbb{N} \). For each \( i \in [j+1, \infty) \) we have \( a_j \leq a_0 \lor a_1 \lor \cdots \lor a_{i-1} \leq b_j \), where the latter inequality holds because of the assumption. Therefore \( a_j \leq \inf_{i \geq j+1} b_i \); this is used in
\[
a_j \land (b_0 \land b_1 \land \cdots \land b_j) \leq (\inf_{i \geq j+1} b_i) \land (b_0 \land b_1 \land \cdots \land b_j) = \inf_{i \in \mathbb{N}} b_i .
\]
This holds for any \( j \in \mathbb{N} \); therefore \( \sup_{j \in \mathbb{N}} (a_j \land (b_0 \land b_1 \land \cdots \land b_j)) \leq \inf_{i \in \mathbb{N}} b_i \). This yields \( \left( \sup_{j \in \mathbb{N}} (a_j \land (b_0 \land b_1 \land \cdots \land b_j)) \right) \lor \inf_{i \in \mathbb{N}} b_i = \inf_{i \in \mathbb{N}} b_i \), which is combined with (24) and proves the claim (16). This concludes the proof of Sublem. A.1. \( \Box \)

We turn back to the proof of Lem. 4.7. By letting \( a_j = \tilde{d} \otimes [\pi^j, \psi_1] \) and \( b_i = \tilde{d} \otimes [\pi^i, \psi_2] \) in Sublem. A.1 we obtain
\[
\inf_{i \in \mathbb{N}} \left( \max_{0 \leq j \leq i-1} \left( \tilde{d} \otimes [\pi^j, \psi_2] \right) \right) = \max_{j \in \mathbb{N}} \left( \min_{0 \leq i \leq j} \left( \tilde{d} \otimes [\pi^i, \psi_1] \right) \right) .
\]
\[ (25) \]

By (15) and (25), we have
\[
\tilde{d} \otimes [\pi, \psi_1 \cup \psi_2] = \max_{j \in \mathbb{N}} \left( \min_{0 \leq i \leq j} \left( \tilde{d} \otimes [\pi^i, \psi_1] \right) \right) .
\]
\[ (26) \]
Let us now look at the value $L(A_{\psi, \varepsilon}^{\psi_1 U \psi_2, \vec{d}})(\pi)$. We analyze possible run trees $\tau$ starting from the state $(\psi_1 U \psi_2, \vec{d})$, much like in the previous case where $|\vec{d}|$ is odd (in the current case it is even). It is easily seen from Def. 4.5 that $\tau$ is of one of the forms shown in Fig. 4:

- If $\tau$ is of the form in Fig. 4 on the left, its utility $\min_{\rho \in \text{path}(\tau)} F^{\infty}(\rho)$ is $\inf_{j \in \mathbb{N}} L(A_{\psi, \varepsilon}^{\psi_2, \vec{d}})(\pi^j)$; note that the rightmost path’s value of $F^{\infty}$ is 1 and hence does not appear here.
- If $\tau$ is of the form in Fig. 4 on the right, its utility $\min_{\rho \in \text{path}(\tau)} F^{\infty}(\rho)$ is given by $\min \{ L(A_{\psi, \varepsilon}^{\psi_1, \vec{d}})(\pi^i), \min_{0 \leq j \leq i} L(A_{\psi, \varepsilon}^{\psi_2, \vec{d}})(\pi^j) \}$ where $i$ is the depth of the last occurrence of the node $(\psi_1 U \psi_2, \vec{d})$.

The value $L(A_{\psi, \varepsilon}^{\psi_1 U \psi_2, \vec{d}})(\pi)$ is defined as the supremum of these utilities. Therefore:

$$L(A_{\psi, \varepsilon}^{\psi_1 U \psi_2, \vec{d}})(\pi) = \max \{ \sup_{i \in \mathbb{N}} \left( \min \left\{ \sup_{j \in \mathbb{N}} \left( \max \left\{ L(A_{\psi_1, \varepsilon, \vec{d}}^{\psi_1, \vec{d}})(\pi^i), \max_{0 \leq j \leq i} L(A_{\psi_2, \varepsilon, \vec{d}}^{\psi_2, \vec{d}})(\pi^j) \right) \right) \right) \right) \}$$

$$= \left\{ \max_{j \in \mathbb{N}} \left( \sup_{i \in \mathbb{N}} \left( \min \left\{ \sup_{j \in \mathbb{N}} \left( \max \left\{ L(A_{\psi_1, \varepsilon, \vec{d}}^{\psi_1, \vec{d}})(\pi^i), \max_{0 \leq j \leq i} L(A_{\psi_2, \varepsilon, \vec{d}}^{\psi_2, \vec{d}})(\pi^j) \right) \right) \right) \right) \right) \right\}$$

$$= \left[ \max_{j \in \mathbb{N}} \left( \sup_{i \in \mathbb{N}} \left( \min \left\{ \sup_{j \in \mathbb{N}} \left( \max \left\{ L(A_{\psi_1, \varepsilon, \vec{d}}^{\psi_1, \vec{d}})(\pi^i), \max_{0 \leq j \leq i} L(A_{\psi_2, \varepsilon, \vec{d}}^{\psi_2, \vec{d}})(\pi^j) \right) \right) \right) \right) \right) \right\}$$

by the induction hypothesis

$$= \left[ \max_{j \in \mathbb{N}} \left( \sup_{i \in \mathbb{N}} \left( \min \left\{ \sup_{j \in \mathbb{N}} \left( \max \left\{ L(A_{\psi_1, \varepsilon, \vec{d}}^{\psi_1, \vec{d}})(\pi^i), \max_{0 \leq j \leq i} L(A_{\psi_2, \varepsilon, \vec{d}}^{\psi_2, \vec{d}})(\pi^j) \right) \right) \right) \right) \right) \right\}$$

concluding the case when $\psi = \psi_1 U \psi_2$ and $|\vec{d}|$ is even.
Suppose that $\psi = \psi_1 \cup_{\eta^+} \psi_2$ and that $|d|$ is odd. We prove the claim by induction on $k$, going backwards, decrementing $k$ starting from the event horizon towards $k = 0$. As the base case, assume that $k$ is big enough and we are beyond the event horizon, that is, $\eta(k) \cdot \prod_{i=1}^n d_i \leq \varepsilon$. Let $\pi \in (\mathcal{P}(AP))^\omega$ and $d \doteq d_1 \ldots d_n$. Then we have $\|\pi, \psi_1 \cup_{\eta^+} \psi_2\| \cdot \prod_{i=0}^n d_i \leq \varepsilon$, by Lem. 2.5 and that $\eta^{+k}(0) = \eta(k)$. It follows from (4) that we have $0 \leq d \otimes \|\pi, \psi\| - d \otimes 0 \leq \varepsilon$ (note that $n = |d|$ is odd). Therefore

$$L(A^{(\psi_1 \cup_{\eta^+} \psi_2, d)}(\pi)) = d \otimes 0 \quad \text{by (6)}$$

$$\in \left[ d \otimes \|\pi, \psi\| - \varepsilon, d \otimes \|\pi, \psi\| \right].$$

Now, as the step case, assume that $\eta(k) \cdot \prod_{i=1}^n d_i > \varepsilon$ and that the claim has been shown for $k + 1$. The analogue below of $\psi_1 \cup \psi_2 \equiv \psi_2 \lor \left(\psi_1 \land X(\psi_1 \cup \psi_2)\right)$ follows easily from Def. 2.4

$$\|\pi, \psi_1 \cup_{\eta^+} \psi_2\| = \max\{ \eta^{+k}(0) \cdot \|\pi, \psi_2\|, \min\{ \eta^{+k}(0) \cdot \|\pi, \psi_1\|, \|\pi^1, \psi_1 \cup_{\eta^{+k+1}} \psi_2\| \} \}.$$  

Therefore

$$d \otimes \|\pi, \psi_1 \cup_{\eta^+} \psi_2\|$$

$$= \max\{ d \otimes (\eta^{+k}(0) \cdot \|\pi, \psi_2\|), \min\{ d \otimes (\eta^{+k}(0) \cdot \|\pi, \psi_1\|), d \otimes \|\pi^1, \psi_1 \cup_{\eta^{+k+1}} \psi_2\| \} \} \quad (27)$$

$$= \max\{ (d \circ \eta^{+k}(0)) \otimes \|\pi, \psi_1\|, d \otimes \|\pi^1, \psi_1 \cup_{\eta^{+k+1}} \psi_2\| \} \},$$

where the first equality is due to the monotonicity of $d \otimes (\_)$, and the second is by (5). Now

$$L(A^{(\psi_1 \cup_{\eta^+} \psi_2, d)}(\pi))$$

$$= \max\{ L(A^{(\psi_2, d \circ \eta^{+k}(0))}(\pi), \min\{ L(A^{(\psi_1, d \circ \eta^{+k}(0))}(\pi), L(A^{(\psi_1 \cup_{\eta^+} \psi_2, d)}(\pi^1) \} \} \} \quad \text{by Def. 2.4}$$

By the induction hypothesis (the claim has been shown for simpler formulas as well as $\psi_1 \cup_{\eta^{+k+1}} \psi_2$), a lower bound of the above value is given by

$$\max\{ \left( (d \circ \eta^{+k}(0)) \otimes \|\pi, \psi_2\| \right) - \varepsilon, \left( (d \circ \eta^{+k}(0)) \otimes \|\pi, \psi_1\| \right) - \varepsilon \}$$

$$= d \otimes \|\pi, \psi_1 \cup_{\eta^+} \psi_2\| - \varepsilon \quad \text{by (27).}$$

Similarly an upper bound $d \otimes \|\pi, \psi_1 \cup_{\eta^+} \psi_2\|$ is obtained by the induction hypothesis and (27). This proves the claim.

The remaining case where $\psi = \psi_1 \cup_{\eta^+} \psi_2$ and $|d|$ is even is similar to the last case. We describe only the base case of induction, where $k$ is big enough so that $\eta(k) \cdot \prod_{i=1}^n d_i \leq \varepsilon$. By Lem. 2.5 we have $\|\pi, \psi\| \in [0, \eta^k(0)]$; therefore

$$0 \leq \eta(k) - \|\pi, \psi\| \leq \eta(k) \leq \varepsilon / \prod_{i=1}^n d_i.$$
By (4) and that \( n \) is even, we have
\[
d\otimes [\pi, \psi] - \tilde{d} \otimes \eta(k) = \left( \prod_{i=1}^{n} d_i \right) \cdot \left( \eta(k) - [\pi, \psi] \right) \in [0, \varepsilon] .
\]

Hence
\[
L(A_{\varphi, \varepsilon}^{(s_1 \cup s_2, \tilde{d})})(\pi) = \tilde{d} \otimes \eta(k) \quad \text{by (6)}
\in \left[ \tilde{d} \otimes [\pi, \psi] - \varepsilon, \tilde{d} \otimes [\pi, \psi] \right] .
\]

This concludes the proof. \( \Box \)

**A.4 Proof of Lem. 4.10**

**Proof.** It follows easily from the definition that there is a bijective correspondence between: a run \( \zeta = (q_0, s_0) \circ (q_1, s_1) \circ \ldots \) of \( A \times K \); and a pair \((\xi, \rho)\) of a path \( \xi = s_0, s_1, \ldots \in \text{path}(K) \) of \( K \) and a run \( \rho \) over \( \lambda(\xi) \) of \( A \). Moreover, the acceptance value of \( \zeta \) in \( A \times K \) is equal to that of \( \rho \) in \( A \). The claim follows immediately. \( \Box \)

**A.5 Proof of Thm. 4.11**

**Proof.**
\[
\langle s_0 s_1 \ldots, \varphi \rangle \geq L(A_{\varphi, \varepsilon})^{na}(\lambda(s_0)\lambda(s_1)\ldots) \quad \text{by Cor. 4.8}
= \max_{\xi \in \text{path}(K)} L(A_{\varphi, \varepsilon})^{na}(\lambda(\xi)) \quad \text{by Lem. 4.10}
\geq \sup_{\xi \in \text{path}(K)} [\xi, \varphi] - \varepsilon \quad \text{by Cor. 4.8}
\]

The solution \( s_0 s_1 \ldots \) thus obtained arises from a lasso computation of \( A_{\varphi, \varepsilon}^{na} \times K \) (by the algorithm in Lem. 3.2), hence is ultimately periodic. \( \Box \)

**A.6 Proof of Prop. 5.1**

**Proof.** Recall that a state of \( A_{\varphi, \varepsilon} \) is a pair \((\psi, \tilde{d})\) of \( \psi \in xcl(\varphi) \) and \( \tilde{d} \in [0, 1]^+ \). We first claim that the number of different \( \psi \)'s is polynomial in \( |\langle \varphi \rangle| \) and \( \log \varepsilon \). The claim is obvious except for the number of the formulas \( \psi \) of the form \( \psi_1 U_{\eta^i} \psi_2, \) for varying \( i \in \mathbb{N} \). Let \( \lambda_0 \) be the maximum number in \( \varphi \) used as the base of an exponential discounting function. For each subformula \( \psi_1 U_{\eta^i} \psi_2 \) of \( \varphi \), the numbers \( i \) for which we have a state \((\psi_1 U_{\eta^i} \psi_2, \tilde{d})\) in \( A_{\varphi, \varepsilon} \) is bounded by \( 1 + \lceil \log \lambda_0 \varepsilon \rceil \). Now we appeal to the fact used in [2] that the value \( \log \lambda_0 \varepsilon = \log \varepsilon / \log \lambda_0 \) is polynomial in the length of the description of \( \lambda_0 \)—hence in \( |\langle \varphi \rangle| \)—and \( \varepsilon \).\(^5\)

\(^5\) It is not explicit in [2] what is meant by the description length of \( \lambda \in (0, 1) \). For the claimed fact to be true—that \( \log \lambda_0 \varepsilon = \log \varepsilon / \log \lambda \) is polynomial in the length of the description of \( \lambda \)—we expect it to be \( a + b \) where \( \lambda = a / b \). For example, when \( \lambda = 1 - \frac{1}{2} \), we have \( \log \lambda_0 \varepsilon = \log \varepsilon - \log(1 - \frac{1}{2}) \leq b \cdot (\log \varepsilon) \) where for the last inequality we used \((\log x)' = \frac{1}{x} \). This is linear in \( b \).
Our second claim is that the number of different \(d\)'s occurring in states of \(A_{\varphi, \varepsilon}\) is exponential in \(|\langle \varphi \rangle|\) and the description length of \(\varepsilon\), hence is the bottleneck in complexity. The length of a discount sequence \(d\) is bounded by the number of negations in \(\varphi\), therefore by \(|\langle \varphi \rangle|\). Each entry \(d_i\) is a multiple \(\lambda_{i_1} \lambda_{i_2} \ldots \lambda_{i_m}\) of different discounting bases \(\lambda_j\) (there are at most \(|\langle \varphi \rangle|\)-many such), and since its value must be bigger than \(\varepsilon\), the length \(m\) of such a multiple is at most \(\log \lambda \varepsilon\). Therefore the number of candidates for \(d_i = \lambda_{i_1} \lambda_{i_2} \ldots \lambda_{i_m}\) is bounded by \(|\langle \varphi \rangle|^{\log \lambda \varepsilon}\); appealing to the fact (see [2]) that \(\log \lambda \varepsilon = \log \varepsilon / \log \lambda\) is polynomial in the length of the description of \(\lambda\) and \(\varepsilon\), we obtain the claim.

\[\square\]

A.7 Proof of Thm. 5.2

Proof. (Sketch) Looking at the construction of Lem. 3.4 in case of \(A = A_{\varphi, \varepsilon}\), we have \(V_Q = \{0, 1\}\), therefore

\[Q' = P(Q \times 2) \times V_\delta \times 2 \cong (P(Q))^2 \times V_\delta \times 2.\]  

(28)

Here the original state space \(Q\) is bounded by \(xcl_\varepsilon(\varphi) \times |\langle \varphi \rangle|^{\log \lambda \varepsilon}\), where

\[xcl_\varepsilon(\varphi) = xcl(\varphi) \setminus \{\varphi_1 U_\eta \varphi_2 \in xcl(\varphi) \mid \eta(0) < \varepsilon\}\]

is a finite set and the second component \(|\langle \varphi \rangle|^{\log \lambda \varepsilon}\) is from the proof of Prop. 5.1.

The optimization lies in the reduction of \(P(Q)\) that occurs in (28) to

\[\left((Q \times Q^2 \times Q^2) \cup \{\bullet\}\right)^{xcl_\varepsilon(\varphi)},\]  

(29)

hence from a double exponential to a single exponential; recall from the proof of Prop. 5.1 that \(Q\) is exponential and \(xcl_\varepsilon(\varphi)\) is polynomial, in \(|\langle \varphi \rangle|\) and the description length of \(\varepsilon\).

The reduction is done concretely as follows. Given a set

\[\{ (\psi, d_1), (\psi, d_2), \ldots, (\psi, d_m) \}\]

(30)

of states of \(Q\) with a common first component \(\psi\), we suppress the set into the function

\[(d_1 \wedge \cdots \wedge d_m) \mathbin{\bigtriangledown} v : v \mapsto \min\{d_1 \boxtimes v, \ldots, d_m \boxtimes v\}\]  

(31)

that does the same job. The latter is a piecewise linear function on \([0, 1]\) and hence is presented as a disjunction of pairs \((f_i, [l_i, r_i])\) of a linear function \(f_i\) and its domain (here \(l_i, r_i \in (0, 1)\)). Now \(f_i\) is represented by some discount sequence so there are at most \(|Q|\)-many of them. A point \(l_i \in [0, 1]\) is expressed as the cross point of two linear functions, each represented by a discount sequence. The same goes for \(r_i\). Moreover, disjunction is taken out of a single state in the resulting automaton—from alternating to non-alternating we only need to bundle up states in conjunction. In summary, to express the piecewise linear function in (31) we need: \(Q\) to represent \(f_i\); \(Q^2\) to represent \(l_i\); and \(Q^2\) to represent \(r_i\), resulting in \(Q \times Q^2 \times Q^2\) in (29).

We consider all those sets in the form of (30), therefore we need \(Q \times Q^2 \times Q^2\) for each formula \(\psi \in xcl_\varepsilon(\varphi)\). The set \(\{\bullet\}\) is in (29) to take care of the case when the set (30) for the formula \(\psi\) is empty.

\[\square\]
B Reduction of Fuzzy Automata to \([0, 1]\)-Acceptance Automata

A generalization of \([0, 1]\)-acceptance automaton is naturally obtained by making transitions also \([0, 1]\)-weighted. The result is called fuzzy automaton and studied e.g. in [16]. Here we show that this generalization does not add expressivity. In fact we prove a more general result, parametrizing \([0, 1]\) into a general semiring \(\mathbb{K}\) (under certain conditions).

We follow [10] and impose certain conditions on a semiring \(K\) of weights.

**Definition B.1 ([10])** A tuple \(\mathbb{K} = (K, \leq, +, \cdot, 0, 1)\) is called an ordered semiring if \((K, +, \cdot, 0, 1)\) is a semiring, \((K, \leq)\) is a partially ordered set and both + and \(\cdot\) are monotonic.

An ordered semiring \(\mathbb{K} = (K, \leq, +, \cdot, 0, 1)\) is said to be lattice-complete if: \((K, \leq)\) is a complete lattice; the units 0, 1 of +, \(\cdot\) satisfy \(0 \leq x \leq 1\) for each \(x \in K\); and

\[
y + \sup_{i \in I} x_i = \sup_{i \in I} (y + x_i)
\]

for each family \((x_i)_{i \in I}\) and each \(y \in K\). We define an infinite sum, as usual, by

\[
\sum_{i \in I} x_i = \sup_{F \in \mathcal{P}_{\text{fin}}(I)} \sum_{i \in F} x_i
\]

where \(\mathcal{P}_{\text{fin}}(I)\) is the set of finite subsets of \(I\).

A semiring is locally finite if the underlying monoid \((K, \cdot, 1)\) is locally finite, that is: for each finite subset \(F \subseteq K\), the submonoid of \((K, \cdot, 1)\) generated by \(F\) is finite.

The notion of \(\mathbb{K}\)-weighted (Büchi) automaton is studied in [10], from which the following definition is taken.

**Definition B.2 (\(\mathbb{K}\)-acceptance (Büchi) automaton, \(\mathbb{K}\)-weighted (Büchi) automaton)**

Let \((K, \leq, +, \cdot, 0, 1)\) be a lattice-complete semiring. A \(\mathbb{K}\)-acceptance (Büchi) automaton is a tuple \(A = (\Sigma, Q, I, \delta, F)\), where \(\Sigma\) is a finite alphabet, \(Q\) is a finite set of states, \(I \subseteq Q\) is a set of initial states, \(\delta : Q \times \Sigma \rightarrow \mathcal{P}(Q)\) is a transition function and \(F : Q \rightarrow K\) is a function that assigns an acceptance value to each state. We define the language \(L(A) : \Sigma^* \rightarrow K\) of \(A\) as

\[
L(A)(w) = \max_{\rho \in \text{run}(w)} \{ F(q) \mid q \in \text{Inf}(\rho) \}.
\]

A \(\mathbb{K}\)-weighted (Büchi) automaton is a tuple \(A = (\Sigma, Q, I, \delta, F)\), where \(\Sigma\) is a finite alphabet, \(Q\) is a finite set of states, \(I : Q \rightarrow K\) is a function assigns an initial weight to each state, \(\delta : Q \times \Sigma \rightarrow K^Q\) is a \(\mathbb{K}\)-weighted transition function and \(F : Q \rightarrow K\) is a function assigns an acceptance value to each state. We define the language \(L(A) : \Sigma^* \rightarrow K\) of \(A\) by

\[
L(A)(w) = \sum_{q_0, q_1, \ldots \in Q^\omega} \inf_{n \in \mathbb{N}} \sup_{i \geq n} \{ I(q_0) \cdot \delta(q_0, w_0)(q_1) \cdot \cdots \cdot \delta(q_{i-1}, w_{i-1})(q_i) \cdot F(q_i) \}.
\]
These notions specialize to $[0, 1]$-acceptance automaton and fuzzy automaton \[16\] by taking the fuzzy semiring $([0, 1], \max, \min, 0, 1)$ as $\mathbb{K}$ in the above definitions.

Locally finiteness of a semiring \[10\] is central in the following result. Its proof is not hard but the result is not explicit in \[10\] or elsewhere.

**Lemma B.3** Let $\mathbb{K} = (K, \leq, +, \cdot, 0, 1)$ be a lattice-complete semiring and $\mathcal{A} = (\Sigma, Q, I, \delta, F)$ be a $\mathbb{K}$-weighted automaton. If $\mathbb{K}$ is locally finite (Def. B.1), there exists a $\mathbb{K}$-acceptance automaton $\mathcal{A}' = (\Sigma, Q', I', \delta', F')$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.

**Proof.** Let $(F, \cdot, 1)$ be the submonoid of $(K, \cdot, 1)$ generated by the (finite) set of weights of transitions occurring in $\mathcal{A}$, that is, $\{\delta(q, a)(q') \mid q, q' \in Q, a \in \Sigma\}$. The set $F$ is finite since $\mathbb{K}$ is locally finite. We now define $\mathcal{A}' = (\Sigma, Q', I', \delta', F')$ as follows.

$$Q' = Q \times F, \quad I' = I \times \{1\},$$

$$\delta'( (q, k), a ) = \{ ( q', k \cdot \delta(q, a)(q') ) \mid q' \in Q \}, \quad F'(q, k) = k \cdot F(q).$$

The proof of $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$ is straightforward. \(\square\)

It is straightforward that the fuzzy semiring $([0, 1], \max, \min, 0, 1)$ is locally finite. This leads to:

**Corollary B.4** Let $\mathcal{A}$ be a fuzzy automaton. There exists a $[0, 1]$-acceptance automaton $\mathcal{A}'$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$. \(\square\)

The main results of \[10, 16\] concern the characterization of so-called $\omega$-rational formal power series over $\mathbb{K}$—those which are generated by $\omega$-regular-like expressions—by $\mathbb{K}$-weighted Büchi automata. Lem. B.3 therefore gives us another characterization by $\mathbb{K}$-acceptance Büchi automata.