Quasi-Weak Cost Automata: A New Variant of Weakness*

Denis Kuperberg[1], **Michael Vanden Boom**[2]

[1]LIAFA/CNRS/Université Paris 7, Denis Diderot, France

[2]Department of Computer Science, University of Oxford, England

ABSTRACT.
Cost automata have a finite set of counters which can be manipulated on each transition but do not affect control flow. Based on the evolution of the counter values, these automata define functions from a domain like words or trees to $\mathbb{N} \cup \{\infty\}$, modulo an equivalence relation which ignores exact values but preserves boundedness properties. These automata have been studied by Colcombet et al. as part of a "theory of regular cost functions", an extension of the theory of regular languages which retains robust equivalences, closure properties, and decidability like the classical theory.
We extend this theory by introducing quasi-weak cost automata. Unlike traditional weak automata which have a hard-coded bound on the number of alternations between accepting and rejecting states, quasi-weak automata bound the alternations using the counter values (which can vary across runs). We show that these automata are strictly more expressive than weak cost automata over infinite trees. The main result is a Rabin-style characterization theorem: a function is quasi-weak definable if and only if it is definable using two dual forms of non-deterministic Büchi cost automata. This yields a new decidability result for cost functions over infinite trees.

1 Introduction

Cost automata are finite-state machines enriched with counters which can be manipulated on each transition but cannot be used to affect control flow. Based on the evolution of the counter values, these automata define functions from some domain (like words or trees over a finite alphabet) to $\mathbb{N} \cup \{\infty\}$, modulo an equivalence relation $\approx$ which ignores exact values but preserves boundedness properties. By only considering the functions up to $\approx$, the resulting "theory of regular cost functions" retains many of the equivalences, closure properties, and decidability results of the theory of regular languages [3]. It extends the classical theory since we can identify each language with its characteristic function mapping structures in the language to 0 and everything else to ∞; it is a strict extension since cost automata can count some behaviour within the input structure.

The development of this theory was motivated by problems which can be reduced to questions of boundedness. For instance, Hashiguchi [7] and later Kirsten [8] used distance and nested-distance desert automata (special forms of cost automata) to prove the decidability of the "star-height problem": given a regular language L of finite words and a natural number n, is there some regular expression (using concatenation, union, and Kleene star) for L which uses at most n nestings of Kleene-star operations? Colcombet and Löding [4] have

*The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement 259454.

used a similar approach over finite trees. The theory of regular cost functions over finite words [3] and finite trees [6] can be viewed as a unifying framework for these problems.

It is desirable to extend the theory to infinite trees. For instance, the "parity-index problem" asks: given a regular language L of infinite trees and $i < j$, is there a parity automaton using only priorities $\{i, i+1, \ldots, j\}$? This is known to be decidable in some special cases (e.g. for deterministic languages [11]), but a general decision procedure is not known. However, Colcombet and Löding [5] have reduced the parity-index problem to another open problem, namely the decidability of $\approx$ for regular cost functions over infinite trees.

Weak cost automata (recently studied in [13]) are a natural starting point in this line of research on regular cost functions over infinite trees. In the classical setting, weak automata are a restricted form of alternating Büchi automata which have a fixed bound on the number of alternations between accepting and rejecting states across all runs. They were introduced in [10] to characterize the languages definable in weak monadic second-order logic (WMSO), a variant of MSO in which second-order quantifiers are interpreted over finite sets. Prior to this work, Rabin [12] had given an interesting characterization using non-deterministic automata, showing that a language is weakly definable if and only if the language and its complement are non-deterministic Büchi recognizable.

These notions of weakness have received considerable attention because the weakly definable languages are expressive (e.g. they capture CTL), but still admit efficient model-checking [9]. Indeed, in order to improve the efficiency in some model-checking scenarios, Kupferman and Vardi [9] adapted Rabin's work and provided a quadratic translation between non-deterministic Büchi automata and the corresponding weak automaton.

In [13], weak cost automata were shown equivalent to "cost WMSO", in analogy to the classical theory. However, the question of a Rabin-style characterization based on non-deterministic automata remained open and prompted this work.

1.1 Contributions

We continue the study of regular cost functions over infinite trees by introducing a variant of weakness which we call "quasi-weakness". Unlike traditional weak automata which have a hard-coded bound on the number of alternations between accepting and rejecting states, quasi-weak automata bound the alternations using counter values (which can vary across runs). We show that quasi-weak cost automata are strictly more expressive than weak cost automata over infinite trees.

Although there is no notion of complement for a function, there are two dual semantics (B and S) used to define cost functions. We show that quasi-weak B-automata can be simulated by non-deterministic B-Büchi and S-Büchi automata. Combined with results from [13], this implies the decidability of $f \approx g$ when f, g are cost functions defined by quasi-weak B-automata. Consequently, this work extends the class of cost functions over infinite trees for which $\approx$ is known to be decidable. We also provide a non-trivial extension of Kupferman and Vardi's construction [9] to translate equivalent non-deterministic B-Büchi and S-Büchi automata to an equivalent quasi-weak B-automaton (where the equivalence in each case is up to $\approx$). This provides a Rabin-style characterization of the functions definable using quasi-weak B-automata and marks an interesting departure from the classical theory.

The construction relies on analyzing a composed run of a B-Büchi automaton and S-Büchi automaton. To aid in this analysis, we use BS-automata and introduce a corresponding BS-equivalence relation $\approxeq$ which can be used to compare cost automata which define not one but two functions (one based on the B-counters and one on the S-counters). Although the B- and S-counter actions in such a composed run can be independent, we show that it is possible to effectively construct an equivalent (up to $\approxeq$) BS-automaton in which the actions are more structured (namely, the counters are "hierarchical" so they can be totally ordered and manipulating a higher counter resets all lower counters). We believe this may be a useful technique in other situations which require both counter types.

1.2 Organization

We define cost automata on infinite trees in Sect. 2, with semantics based on two-player infinite games. We also introduce the new quasi-weak automata and compare to more traditional weak automata. In Sect. 3, we consider automata with both counter types and show how they can be made hierarchical. Finally, in Sect. 4, we describe the other components of the main result, a Rabin-style characterization for quasi-weak cost automata. We conclude with some open questions in Sect. 5.

1.3 Notations

We write $\mathbb{N}$ for the set of non-negative integers and $\mathbb{N}_\infty$ for the set $\mathbb{N} \cup \{\infty\}$, ordered by $0 < 1 < \cdots < \infty$. If $i \leq j$ are integers, $[i,j]$ denotes the set $\{i, i+1, \ldots, j\}$. We fix a finite alphabet $\mathbb{A}$. The set of finite (respectively, infinite) words over $\mathbb{A}$ is $\mathbb{A}^*$ (respectively, $\mathbb{A}^\omega$) and the empty word is ϵ. For notational simplicity we work only with infinite binary trees. Let $\mathcal{T} = \{0,1\}^*$ be the unlabelled infinite binary tree. A *branch* in $\mathcal{T}$ is a word $\pi \in \{0,1\}^\omega$. The set $\mathcal{T}_\mathbb{A}$ of complete $\mathbb{A}$-labelled binary trees is composed of mappings $t : \mathcal{T} \to \mathbb{A}$.

Non-decreasing functions $\mathbb{N} \to \mathbb{N}$ will be denoted by letters $\alpha, \beta, \ldots$, and will be extended to $\mathbb{N}_\infty$ by $\alpha(\infty) = \infty$. We call these *correction functions*.

2 Cost Automata

2.1 Cost Functions

Let E be any set, and $\mathcal{F}_E$ be the set of functions $: E \to \mathbb{N}_\infty$. For $f, g \in \mathcal{F}_E$ and α a correction function, we write $f \preccurlyeq_\alpha g$ if $f \leq \alpha \circ g$ (or if we are comparing single values $n, m \in \mathbb{N}$, $n \preccurlyeq_\alpha m$ if $n \leq \alpha(m)$). We write $f \approx_\alpha g$ if $f \preccurlyeq_\alpha g$ and $g \preccurlyeq_\alpha f$. Finally, $f \approx g$ (respectively, $f \preccurlyeq g$) if $f \approx_\alpha g$ (respectively, $f \preccurlyeq_\alpha g$) for some α. The idea is that the *boundedness relation* $\approx$ does not pay attention to exact values, but does preserve the existence of bounds. Remark that $f \not\approx g$ if and only if there exists a set $D \subseteq E$ such that g is bounded on D but f is unbounded on D.

A *cost function over* E is an equivalence class of $\mathcal{F}_E/\approx$. In practice, a cost function (denoted $f, g, \ldots$) will be represented by one of its elements in $\mathcal{F}_E$. In this paper, E will usually be $\mathcal{T}_\mathbb{A}$. The functions defined by automata will always be considered as cost functions, i.e. only considered up to $\approx$.

2.2 *B*- and *S*-Valuations

Cost automata define functions from $\mathcal{T}_{\mathbb{A}}$ to $\mathbb{N}_\infty$. The valuation is based on both the classic Büchi acceptance condition and a finite set of counters Γ.

A counter γ is initially assigned value 0 and can be *incremented* $\mathtt{i}$, *reset* $\mathtt{r}$ to 0, *checked* $\mathtt{c}$, or left unchanged ε. Given an infinite word u_γ over the alphabet $\{\varepsilon, \mathtt{i}, \mathtt{r}, \mathtt{c}\}$, we define a set $C(u_\gamma) \subseteq \mathbb{N}$ which collects the checked values of γ. In the case of a finite set of counters Γ and a word u over $\{\varepsilon, \mathtt{i}, \mathtt{r}, \mathtt{c}\}^\Gamma$, $C(u) := \bigcup_{\gamma\in\Gamma} C(pr_\gamma(u))$ ($pr_\gamma(u)$ is the γ-projection of u).

We will separate counters into two types: *B*-counters, which accept as atomic actions the set $\mathbb{B} = \{\varepsilon, \mathtt{ic}, \mathtt{r}\}$, and *S*-counters, with atomic actions $\mathbb{S} = \{\varepsilon, \mathtt{i}, \mathtt{r}, \mathtt{cr}\}$. Given *B*-counters Γ_B and $u \in (\mathbb{B}^{\Gamma_B})^\omega$, the *B-valuation* is $val_B(u) := \sup C(u)$; likewise, given *S*-counters Γ_S and $u \in (\mathbb{S}^{\Gamma_S})^\omega$, the *S-valuation* is $val_S(u) := \inf C(u)$. By convention, $\inf\emptyset = \infty$ and $\sup\emptyset = 0$. For instance $val_B((\mathtt{ic})^\omega) = \infty$, $val_B((\mathtt{icr})^\omega) = 1$, $val_S(\mathtt{i}^{100}\mathtt{cri}\varepsilon\mathtt{icr}(\mathtt{r})^\omega) = 2$, and $val_S(\mathtt{i}^\omega) = \infty$ because the counter is never checked.

In all cases, if the set of counters Γ is $[1, k]$, an action ν is called *hierarchical* if there is some $i \in [1, k]$ such the action ν performs ε on all counters $j > i$, and $\mathtt{r}$ on all counters $j < i$. It means that performing an increment or a reset on counter i resets all counters j below it.

Cost automata are named *B*-, *S*-, or *BS*-automata depending on the type(s) of counters used. They are *hierarchical* (written, e.g. *hB*-automata) if only hierarchical actions are used.

2.3 *B*- and *S*-Automata on Infinite Trees

An *alternating B-Büchi automaton* $\mathcal{A} = \langle Q, \mathbb{A}, q_0, F, \Gamma_B, \delta\rangle$ on infinite trees has a finite set of states Q, alphabet $\mathbb{A}$, initial state $q_0 \in Q$, accepting states F, finite set Γ_B of *B*-counters, and transition function $\delta : Q \times \mathbb{A} \to \mathcal{B}^+(\{0,1\} \times \mathbb{B}^{\Gamma_B} \times Q)$, where $\mathcal{B}^+(\{0,1\} \times \mathbb{B}^{\Gamma_B} \times Q)$ is the set of positive boolean combinations, written as a disjunction of conjunctions of elements $(d, \nu, q) \in \{0,1\} \times \mathbb{B}^{\Gamma_B} \times Q$. *Alternating S-Büchi automata* are defined in the same way, replacing *B*-counters by *S*-counters and $\mathbb{B}$ with $\mathbb{S}$.

We view running a *B*-automaton (resp. *S*-automaton) $\mathcal{A}$ on an input tree t as a game $(\mathcal{A}, t)$ between two players : Eve is in charge of the disjunctive choices and tries to minimize (resp. maximize) counter values while satisfying the Büchi condition, and Adam is in charge of the conjunctive choices and tries to maximize (resp. minimize) counter values or show the Büchi condition is not satisfied. Because the transition function is given as a disjunction of conjunctions, we can consider that at each position, Eve first chooses a disjunct, and then Adam chooses a single tuple (d, ν, q) in this disjunct.

A *play* of $\mathcal{A}$ on input t is a sequence $q_0, (d_1, \nu_1, q_1), (d_2, \nu_2, q_2), \ldots$ compatible with t and δ, i.e. q_0 is initial, and for all $i \in \mathbb{N}$, $(d_{i+1}, \nu_{i+1}, q_{i+1})$ appears in $\delta(q_i, t(d_1 \ldots d_i))$.

A *strategy* for Eve (resp. Adam) in the game $(\mathcal{A}, t)$ is a function that fixes the next choice of Eve (resp. Adam), based on the history of the play (resp. the history of the play and Eve's choice of disjunct). A strategy is *finite-memory* if the number of memory states needed for the player to choose the next move is finite. A strategy is *positional* if no memory at all is needed: the player only needs to know the current position. Notice that choosing a strategy for Eve and a strategy for Adam fixes a play in $(\mathcal{A}, t)$. We say a play π is *compatible* with a strategy σ for Eve if there is some strategy σ' for Adam such that σ and σ' yield the play π.

A play π is *accepting* if there is $q \in F$ appearing infinitely often in π (i.e. π satisfies the Büchi acceptance condition). Given a play π from a B-automaton $\mathcal{A}$, the value of π is $val(\pi) := val_B(h_B(\pi))$ if π is accepting, and $val(\pi) = \infty$ otherwise (where h_B is the projection of π to the B-actions). This yields the maximum checked counter value if the play is accepting, and ∞ otherwise. We assign a value to a strategy σ for Eve by $val(\sigma) := \sup\{val(\pi) : \pi \text{ is compatible with } \sigma\}$. The value of $\mathcal{A}$ over a tree t is $[\![\mathcal{A}]\!]_B(t) := \inf\{val(\sigma) : \sigma \text{ is a strategy for Eve in the game } (\mathcal{A}, t)\}$.

Likewise, in an S-automaton $\mathcal{A}'$, we define $val(\pi) := val_S(h_S(\pi))$ if π is accepting, and 0 otherwise (where h_S is the projection to the S-actions). Once again, counter actions are only considered if the play is accepting (this time the minimum checked value is used), and 0 is assigned to rejecting plays. Then $val(\sigma) := \inf\{val(\pi) : \pi \text{ is compatible with } \sigma\}$, and $[\![\mathcal{A}']\!]_S(t) := \sup\{val(\sigma) : \sigma \text{ is a strategy for Eve in the game } (\mathcal{A}', t)\}$.

We consider $[\![\mathcal{A}]\!]_B$ and $[\![\mathcal{A}']\!]_S$ as cost functions, so we always work modulo the cost function equivalence $\approx$. If it is clear what semantic the automaton uses we will omit the subscript and write just $[\![\mathcal{A}]\!]$ or $[\![\mathcal{A}']\!]$. If $f \approx [\![\mathcal{A}]\!]$ then we say $\mathcal{A}$ *recognizes* the cost function f.

If for all $(q, a) \in Q \times \mathbb{A}$, $\delta(q, a)$ is of the form $\bigvee_i (0, \nu_i, q_i) \wedge (1, \nu_i', q_i')$, then we say the automaton is *non-deterministic*. We define a *run* to be the set of possible plays compatible with some fixed strategy of Eve. Since the only choices of Adam are in the branching, a run labels the entire binary tree with states, and choosing a branch yields a unique play of the automaton. A run is accepting if all of its plays are accepting (that is, if it is accepting on all branches). A value is assigned to a run of a B-automaton (resp. S-automaton) by taking the supremum (resp. infimum) of the values across all branches.

Finally, a cost automaton $\mathcal{A} = \langle Q, \mathbb{A}, q_0, F, \Gamma, \delta \rangle$ is *weak* if the state-set Q can be partitioned into $Q_1, \dots, Q_k$ satisfying:

- for all i and for all $q, q' \in Q_i$, $q \in F$ if and only if $q' \in F$;
- if some (d, ν, q) appears in some $\delta(p, a)$ with $p \in Q_i$ and $q \in Q_j$, then $j \leq i$.

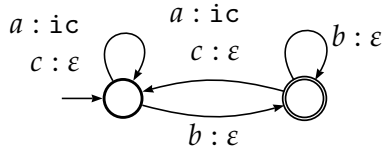
This means there is a fixed bound k on the number of alternations between accepting and rejecting states, so any accepting play must stabilize in an accepting partition.

Examples

Let $\mathbb{A} = \{a, b, c\}$ and f be the cost function over $\mathbb{A}$-labelled trees where $f(t) = \infty$ if there is a branch with only finitely many b's, and $f(t) = \sup\{|\pi|_a : \pi \text{ is a branch of } t\}$ otherwise, where $|\pi|_a$ denotes the number of a's in π.

We define a non-deterministic B-Büchi automaton $\mathcal{U}$ and a non-deterministic S-Büchi automaton $\mathcal{U}'$, together with a weak automaton $\mathcal{B}$, such that $f \approx [\![\mathcal{U}]\!] \approx [\![\mathcal{U}']\!] \approx [\![\mathcal{B}]\!]$.

The principle of $\mathcal{U}$ is to simultaneously count a's and check that infinitely many b's are seen by running the following deterministic B-automaton on every branch. We write $a : \nu$ to denote that on input a, the counter action is ν; accepting states are denoted double circles.

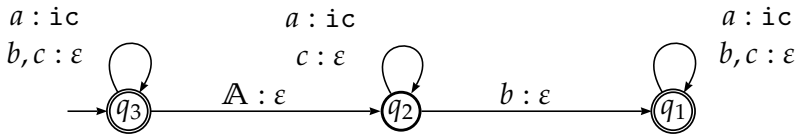


On the other hand, $\mathcal{U}' = \langle \{p_a, p_b, q_b, \top\}, \mathbb{A}, \{p_a, p_b\}, \{q_b, \top\}, \{\gamma\}, \delta \rangle$ tries to find a branch π with either a lot of a's (state p_a), or only finitely many b's (state p_b), in order to witness a high value for f (∞ in the second case). For simplicity, we allow here two initial states, but this does not add expressive power to the model. The state q_b is used when Eve has guessed the position of the last b, and still needs to prove that there are no more b on π, and $\top$ is used when the remainder of the branch does not matter.

The transition table δ for $\mathcal{U}'$ follows. When the counter action is missing, it means that it can be either ε or $\mathtt{r}$, because the counter values no longer matter. Remark that $\mathcal{U}'$ is in fact a non-deterministic weak S-automaton.

δ	p_a	p_b	q_b	$\top$
a	$((0,\mathtt{i},p_a)\wedge(1,\top))$ $\vee((0,\top)\wedge(1,\mathtt{i},p_a))$ $\vee((0,\mathtt{cr},\top)\wedge(1,\top))$ $\vee((0,\top)\wedge(1,\mathtt{cr},\top))$	$((0,p_b)\wedge(1,\top))$ $\vee((0,\top)\wedge(1,p_b))$ $\vee((0,q_b)\wedge(1,\top))$ $\vee((0,\top)\wedge(1,q_b))$	$((0,q_b)\wedge(1,\top))$ $\vee((0,\top)\wedge(1,q_b))$	$(0,\top)\wedge(1,\top)$
b	$((0,\varepsilon,p_a)\wedge(1,\top))$ $\vee((0,\top)\wedge(1,\varepsilon,p_a))$ $\vee((0,\mathtt{cr},\top)\wedge(1,\top))$ $\vee((0,\top)\wedge(1,\mathtt{cr},\top))$	$=\delta(p_b,a)$	*false*	$(0,\top)\wedge(1,\top)$
c	$=\delta(p_a,b)$	$=\delta(p_b,a)$	$=\delta(q_b,a)$	$(0,\top)\wedge(1,\top)$

Finally, $\mathcal{B}$ is designed such that Adam controls all of the choices: Adam selects a single branch, and runs the following automaton on this branch (he controls the non-determinism):



If there is a branch π with finitely many b's, Adam can select π and stabilize in rejecting state q_2 by moving from q_1 to q_2 after the last b. This witnesses value ∞ for f. Otherwise, Adam tries to select a branch which maximizes the number of a's. The state-set can be partitioned such that $Q_i = \{q_i\}$ for $i \in [1,3]$.

2.4 Quasi-Weak B-Automata

We want to define an extension of weak B-automata, which preserves the property that accepting plays must stabilize in accepting states. The idea of weak automata is to bound the number of alternations between accepting and rejecting states by a hard bound.

Here we have another available tool to bound the number of such alternations: the counters. We know that in a B-automaton, an accepting play of finite value n does at most n increments between resets, but this number is not known a priori by the automaton. Thus, if we guarantee there is correction function α such that in any play π of value n, $\alpha(n)$ is greater than the number of alternations between accepting and rejecting states in π, then we know that any play of finite value must stabilize in accepting states. Otherwise, infinitely many alternations would give value ∞ to the cost function computed by the automaton.

Thus we define quasi-weak automata in the following way:

DEFINITION 1. *An alternating B-Buchi automaton is quasi-weak if there is a correction function α such that in any play of $\mathcal{A}$ of value $n < \infty$, the number of alternations between accepting and rejecting states is smaller than $\alpha(n)$.*

In particular, any weak automaton $\mathcal{A}$ is quasi-weak since we can take $\alpha(n) = k$ for all n, where k is the number of partitions of $\mathcal{A}$. We can also give a structural characterization.

PROPOSITION 2. *An alternating B-Büchi automaton is quasi-weak if and only if in any reachable cycle containing both accepting and rejecting states, some counter is incremented but not reset.*

We say a cost function is *quasi-weak* if it is recognized by some quasi-weak B-automaton.

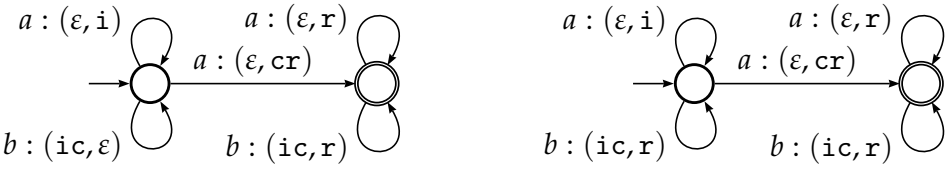
PROPOSITION 3. *There exists a cost function over infinite trees which is recognized by a non-deterministic quasi-weak B-automaton, but not by any weak B-automaton. Consequently, quasi-weak B-automata are strictly more expressive than weak B-automata.*

PROOF. (Sketch) The idea is to build an explicit cost function f, and for each $n \in \mathbb{N}$ an infinite tree t_n which includes labels that dictate which player controls each position in the game (this is inspired by [1]). These trees are designed such that any alternating B-automaton recognizing f is forced to do $\Theta(n)$ alternations between accepting and rejecting states on t_n. This shows f cannot be computed by a weak B-automaton. On the other hand, we give an explicit non-deterministic quasi-weak B-automaton for f.

3 BS-Automata

We usually work with cost automata with only one type of counter, B or S. In the next section, however, we compose runs from B-Büchi and S-Büchi automata and consequently must work with both counter types simultaneously. We capture this in a *non-deterministic BS-Büchi automaton* $\mathcal{A} = \langle Q, \mathbb{A}, q_0, F_B, F_S, \Gamma_B, \Gamma_S, \Delta \rangle$. Such an automaton defines functions $[\![\mathcal{A}]\!]_B$ and $[\![\mathcal{A}]\!]_S$ as expected (by restricting to one of the counter types).

Let $\mathcal{A}$ and $\mathcal{A}'$ be the following non-deterministic BS-automata on infinite words over $\mathbb{A} := \{a, b, c\}$, each with one B- and one S-counter. We write $a : (d, d')$ if on input a, the output is action d (resp. d') for the B (resp. S) counter. We omit self-loops $c : (\varepsilon, \varepsilon)$.



These automata are very similar. For instance, $[\![\mathcal{A}]\!]_B = [\![\mathcal{A}']\!]_B = |\cdot|_b$. The key difference is $\mathcal{A}'$ is *hierarchical*, with the B-counter above the S-counter. Formally, the counters $\Gamma_B \uplus \Gamma_S$ are globally numbered $[1, k]$ (for $k = |\Gamma_B| + |\Gamma_S|$) and for any action on $\mathbb{B}^{\Gamma_B} \times \mathbb{S}^{\Gamma_S}$ there is some $i \in [1, k]$ such that ε is performed on all counters $j > i$ and $\mathtt{r}$ on all counters $j < i$.

Notice that we have $[\![\mathcal{A}]\!]_S \approx |\cdot|_a$ (if there are a finite number of a's, then the best run of $\mathcal{A}$ moves to the accepting state when reading the final a; otherwise, for every n, there is an accepting run of $\mathcal{A}$ such that the S-counter has value n). In $\mathcal{A}'$, however, the B-counter is higher than the S-counter so $\mathcal{A}'$ forces a reset of the S-counter when a b is read in the

initial state. Since there is no a priori bound on the number of b's in the input, this means $[\![\mathcal{A}']\!]_S \not\approx [\![\mathcal{A}]\!]_S$. However, for any fixed m and any u such that $[\![\mathcal{A}]\!]_B(u) \leq m$, the S-value of $\mathcal{A}$ on u is $\approx_{\beta_m}$-equivalent to $\mathcal{A}'$ on u with $\beta_m(n) = n(m+1)$.

This motivates a new equivalence relation $\cong$ which we call *BS-equivalence*. We define $\mathcal{A} \cong \mathcal{A}'$ to hold if (i) $[\![\mathcal{A}]\!]_B \approx_\alpha [\![\mathcal{A}']\!]_B$ and (ii) for any m, there is a correction function β_m such that the S-values of $\mathcal{A}$ and $\mathcal{A}'$ are $\approx_{\beta_m}$-equivalent when restricted to inputs with B-values at most $\alpha(m)$. Although it is technical, this definition captures the notion that two BS-Büchi automata behave in a similar fashion (as in the example above).

It turns out that given any BS-automaton like $\mathcal{A}$, there is an hBS-automaton $\mathcal{A}'$ satisfying $\mathcal{A} \cong \mathcal{A}'$. Moreover, this translation can be done effectively by transducers which read an infinite word of non-hierarchical counter actions and output hierarchical counter actions. This is in analogy to the deterministic transducer which can be used to translate a Muller condition to a parity condition in the classical setting, or the transducer defined in [6] which translates B-actions to hierarchical B-actions. A similar idea is also used in [2] for automata with both B- and S-counters but in a setting where only boolean properties about boundedness and unboundedness are considered (unlike the quantitative setting here).

THEOREM 4. *For all sets Γ_B, Γ_S of counters, there exists effectively a history-deterministic hBS-automaton $\mathcal{H}(\Gamma_B, \Gamma_S)$ on infinite words over $\mathbb{B}^{|\Gamma_B|} \times \mathbb{S}^{|\Gamma_S|}$ with $\mathcal{H}(\Gamma_B, \Gamma_S) \cong \mathcal{G}(\Gamma_B, \Gamma_S)$ where $\mathcal{G}(\Gamma_B, \Gamma_S)$ is the BS-automaton which copies the counter actions from the input.*

The transducer $\mathcal{H}(\Gamma_B, \Gamma_S)$ has the same set of B counters, but extra copies of the S-counters. The principle of the automaton is to split the input word into sequences of S-actions from $\{\mathtt{i}, \varepsilon\}^*$ which are between resets of the B-counters. It uses one copy of the S-counter to count the number of S-increments within each sequence, and another copy to count the sequences with at least one S-increment. If the S-value is high compared to the B-value, then the transducer will also have a high S-value, obtained from one of the copies.

These transducers are *history-deterministic*, a weakening of traditional determinism [3]. The entire history of the input and the current state are required to determine the next transition (rather than just the current state and input letter). Because the choice of the transition depends only on the past, for any two input words, the automaton can find good moves which do not conflict on any common prefix. This means these automata (like deterministic automata) compose well with alternating automata and games: they can be simulated on each play in a game while preserving the value up to $\approx$ or $\cong$ (see [3] for more information).

This means that we can transform arbitrary BS-automata over words or trees into hierarchical BS-automata which are easier to work with.

4 Characterization of Quasi-Weak Cost Automata

In this section we prove a Rabin-style characterization for quasi-weak B-automata:

THEOREM 5. *A cost function f over infinite trees is recognizable by some quasi-weak B-automaton $\mathcal{B}$ if and only if there is a non-deterministic B-Büchi automaton $\mathcal{U}$ and non-deterministic S-Büchi automaton $\mathcal{U}'$ such that $f \approx [\![\mathcal{U}]\!]_B \approx [\![\mathcal{U}']\!]_S$.*

The first direction is described in Lemmas 6 and 7 in Section 4.1. The other direction is described in Sections 4.2–4.4, culminating in Theorem 10.

4.1 Simulation

We start by showing that a quasi-weak B-automaton (in fact, any alternating B-Büchi automaton) $\mathcal{A}$ can be simulated by a non-deterministic B-Büchi version.

LEMMA 6. *Given an alternating B-automaton $\mathcal{B}$, a non-deterministic B-Büchi automaton $\mathcal{U}$ can be effectively constructed such that $[\![\mathcal{B}]\!]_B \approx [\![\mathcal{U}]\!]_B$.*

PROOF. (Sketch) Recall that in a B-Büchi game, the value of a strategy is the maximum over all plays compatible with the strategy. Hence, we first show there is a B-Büchi automaton $\mathcal{D}_{\max}$ recognizing $\mathit{max\text{-}play}(w_\sigma^\tau) = \sup\{\mathit{val}(\pi) : \pi \text{ is compatible with } \sigma \text{ and stays on } \tau\}$ on words w_σ^τ. We cannot guarantee that $\mathcal{D}_{\max}$ is deterministic, but it is history-deterministic (which means it can be simulated over each branch like a deterministic automaton).

On input t, the non-deterministic B-Büchi $\mathcal{U}$ guesses annotations yielding some t_σ, checks the annotations describe a valid strategy σ in $(\mathcal{B}, t)$, and simulates $\mathcal{D}_{\max}$ on each branch in t_σ (possible since $\mathcal{D}_{\max}$ is history-deterministic). Because non-determinism resolves into taking an infimum, $\mathcal{U}$ calculates the infimum over the values of all finite-memory strategies in $(\mathcal{B}, t)$. Although finite-memory strategies might not achieve the optimal value, they do achieve an $\approx$-equivalent value in B-Büchi games by [13]. Hence, $[\![\mathcal{B}]\!] \approx [\![\mathcal{U}]\!]$.

Proving that $\mathcal{B}$ can be simulated by a non-deterministic S-Büchi automaton $\mathcal{U}'$ is more technical and uses the fact that $\mathcal{B}$ is quasi-weak.

LEMMA 7. *Given a quasi-weak B-automaton $\mathcal{B}$, a non-deterministic S-Büchi automaton $\mathcal{U}'$ can be effectively constructed such that $[\![\mathcal{B}]\!]_B \approx [\![\mathcal{U}']\!]_S$.*

PROOF. (Sketch) The automaton $\mathcal{U}'$ can no longer guess a strategy in $(\mathcal{B}, t)$, since the value of $(\mathcal{B}, t)$ is the infimum over all strategies and non-determinism in an S-automaton resolves into taking a supremum. Instead, we consider a dual game $\overline{(\mathcal{B}, t)}$ where the roles of the players are reversed so Eve tries to maximize the B-value across all strategies. We show there is a history-deterministic S-Büchi automaton $\mathcal{D}_{\min}$ which computes the minimum value of a set of plays from such a game. We show these games admit finite-memory strategies. The S-Büchi automaton $\mathcal{U}'$ guesses a finite-memory strategy in such a game and then simulates $\mathcal{D}_{\min}$ on each branch of the tree annotated with this strategy in order to compute its value.

These simulation lemmas and [13, Lemma 1] imply a new decidability result (extending the class of cost functions over infinite trees for which decidability of $\approx$ is known).

COROLLARY 8. *If f, g are cost functions over infinite trees which are given by quasi-weak B-automata then it is decidable whether or not $f \preccurlyeq g$.*

4.2 Construction from Kupferman and Vardi

We now turn to the other direction of Theorem 5. The corresponding classical result states that given non-deterministic Büchi automata $\mathcal{U}$ and $\mathcal{U}'$ such that $L(\mathcal{U})$ is the complement of $L(\mathcal{U}')$, there is a weak automaton $\mathcal{A}$ such that $L(\mathcal{A}) = L(\mathcal{U})$ [9].

The proofs in [12, 9] begins with an analysis of composed runs of $\mathcal{U}$ and $\mathcal{U}'$. Let $m := |Q| \cdot |Q'|$. A *frontier* E is a set of nodes of t such that for any branch π of t, $E \cap \pi$ is a singleton. Kupferman and Vardi [9] define a *trap* for $\mathcal{U}$ and $\mathcal{U}'$ to be a strictly increasing sequence of

frontiers $E_0 = \{\epsilon\}, E_1 \ldots, E_m$ such that there exists a tree t, a run R of $\mathcal{U}$ on t, and a run R' of $\mathcal{U}'$ on t satisfying the following properties: for all $0 \leq i < m$ and for all branches π in t, there exists $x, x' \in [e_i^\pi, e_{i+1}^\pi)$ such that $R(x) \in F$ and $R'(x') \in F'$ where $e_0^\pi < \cdots < e_m^\pi$ is the set of nodes from $E_0, \ldots, E_m$ induced by π. The set of positions $[e_i^\pi, e_{i+1}^\pi)$ can be viewed as a block, and each block in a trap witnesses an accepting state from $\mathcal{U}$ and $\mathcal{U}'$.

This is called a trap because $L(\mathcal{U}')$ is the complement of $L(\mathcal{U})$, but a trap implies $L(\mathcal{U}) \cap L(\mathcal{U}') \neq \emptyset$ (using a pumping argument on blocks). The weak automaton $\mathcal{A}$ has Eve (resp. Adam) select a run of $\mathcal{U}$ (resp. $\mathcal{U}'$). The acceptance condition requires that any time an accepting state from $\mathcal{U}'$ is seen, an accepting state from $\mathcal{U}$ is eventually seen. Because of the trap condition, these accepting blocks only need to be counted up to m times (so $\mathcal{A}$ is weak).

4.3 Cost Traps

Now let $\mathcal{U} = \langle Q_\mathcal{U}, \mathbb{A}, q_0^\mathcal{U}, F_B^\mathcal{U}, \Gamma_B^\mathcal{U}, \Delta_\mathcal{U} \rangle$ (respectively, $\mathcal{U}' = \langle Q_{\mathcal{U}'}, \mathbb{A}, q_0^{\mathcal{U}'}, F_S^{\mathcal{U}'}, \Gamma_S^{\mathcal{U}'}, \Delta_{\mathcal{U}'} \rangle$) be a non-deterministic B-Büchi (respectively, S-Büchi) automaton such that $[\![\mathcal{U}]\!]_B \approx [\![\mathcal{U}']\!]_S$. Our goal is to construct a quasi-weak B-automaton $\mathcal{B}$ which is equivalent to $\mathcal{U}$.

We want to extend the classical case to cost functions, so we seek a notion of "cost trap", which will imply a contradiction with $[\![\mathcal{U}]\!]_B \approx [\![\mathcal{U}']\!]_S$. More specifically, we want a notion of blocks and traps which will witness a bounded B-value from $\mathcal{U}$ on some set of trees but an unbounded S-value for $\mathcal{U}'$ on the same set (showing $[\![\mathcal{U}']\!]_S \not\preccurlyeq [\![\mathcal{U}]\!]_B$). The definition of a block when using arbitrary B- and S-counter actions coming from $\mathcal{U}$ and $\mathcal{U}'$ would be very intricate because it would have to deal with the interaction of the B- and S-actions. In order to avoid this, we switch to working with a non-deterministic hBS-Büchi automaton $\mathcal{A} = \langle Q_\mathcal{A}, \mathbb{A}, q_0^\mathcal{A}, F_B, F_S, \Gamma_B, \Gamma_S, \delta_\mathcal{A} \rangle$ which is BS-equivalent to $\mathcal{U} \times \mathcal{U}' = \langle Q_\mathcal{U} \times Q_{\mathcal{U}'}, \mathbb{A}, (q_0^\mathcal{U}, q_0^{\mathcal{U}'}), F_B^\mathcal{U}, F_S^{\mathcal{U}'}, \Gamma_B^\mathcal{U}, \Gamma_S^{\mathcal{U}'}, \Delta_{\mathcal{U}\times\mathcal{U}'} \rangle$ but uses hierarchical counters.

A *block* based on hierarchical BS-actions from $\mathcal{A}$ has accepting states from both F_B and F_S (corresponding to accepting states for $\mathcal{U}$ and $\mathcal{U}'$), but it also has a reset for B-counter γ if γ is incremented in that block (in order to ensure pumping does not inflate the B-value). The number of blocks required is also increased to $m := (|Q_\mathcal{A}| + 2)^{|\Gamma_S|+1}$ for technical reasons.

A *cost trap* for $\mathcal{A}$ is a frontier E_m and for every branch π up to E_m a strictly increasing set of nodes $e_0^\pi < \cdots < e_m^\pi \in E_m$ such that there exists a tree t and a run R of $\mathcal{A}$ on t with $val_S(R) > |Q_\mathcal{A}|$ satisfying the following properties: for all $0 \leq i < m$ and for all branches π, $[e_i^\pi, e_{i+1}^\pi)$ is a block; if branches π_1 and π_2 share some prefix up to position y and $x < y$ is the first position with $e_i^{\pi_1} = x$ and $e_i^{\pi_2} \neq x$ then $e_i^{\pi_2} > y$ (i.e. pumping blocks from π_2 does not damage blocks from π_1).

A pumping argument shows a cost trap implies $\mathcal{U}$ and $\mathcal{U}'$ are not equivalent.

PROPOSITION 9. *Let $\mathcal{U}$ (respectively, $\mathcal{U}'$) be non-deterministic B-Büchi (respectively, S-Büchi). Let $\mathcal{A} \cong \mathcal{U} \times \mathcal{U}'$ be a non-deterministic hBS-automaton. If there exists a cost trap for $\mathcal{A}$, then $[\![\mathcal{U}']\!] \not\preccurlyeq [\![\mathcal{U}]\!]$.*

4.4 Construction of Quasi-Weak B-Automaton $\mathcal{B}$

Given $\mathcal{U}$ and $\mathcal{U}'$ with $[\![\mathcal{U}]\!]_B \approx [\![\mathcal{U}']\!]_S$, we can effectively build a quasi-weak B-automaton $\mathcal{B}$ which on an input tree t,

- simulates in parallel $\mathcal{U}$ (driven by Eve) and $\mathcal{U}'$ (driven by Adam) over t;
- runs the hBS-transducer $\mathcal{H}(\Gamma_B^{\mathcal{U}}, \Gamma_S^{\mathcal{U}'})$ over the composed actions from $\mathcal{U}$ and $\mathcal{U}'$;
- analyzes the output of this transducer together with the accepting states of $\mathcal{U}$ and $\mathcal{U}'$, keeping track of blocks in order to build a cost trap;
- outputs the B-actions of $\mathcal{U}$.

The key difference from the classical case is in the block counting. In [9], the block number only increases and it suffices to count up to a fixed bound. Since each block contains at most 2 alternations between accepting and rejecting states, this results in a weak automaton.

Here, we also have to forbid in any block the presence of an increment for some counter γ without a reset for γ. However, it may be the case that on a branch of a run of $\mathcal{U}$ some counter is incremented but is never reset. So the automaton $\mathcal{B}$ may start counting blocks only to have to restart the counting if an increment is seen which does not have a later reset. But this means that any decrease in the block number corresponds to an increase in the cost of the play. Hence, the bound on the number of alternations depends on the value of the automaton, which is exactly the property of a quasi-weak automaton.

The idea for the proof that $[\![\mathcal{B}]\!]_B \approx [\![\mathcal{U}]\!]_B \approx [\![\mathcal{U}']\!]_S$ is that if $\mathcal{U}$ accepts some t with low value, then it gives Eve a strategy of the same value in $(\mathcal{B}, t)$. On the other hand, assuming (for the sake of contradiction) that Eve has a low-value strategy in $(\mathcal{B}, t)$ but $\mathcal{U}$ actually assigns t a high value results in a cost trap, which is absurd. Hence, we get the main result:

THEOREM 10. *If there is a non-deterministic B-Büchi automaton $\mathcal{U}$ and a non-deterministic S-Büchi automaton $\mathcal{U}'$ such that $[\![\mathcal{U}]\!]_B \approx [\![\mathcal{U}']\!]_S$, then we can effectively construct a quasi-weak alternating B-automaton $\mathcal{B}$ such that $[\![\mathcal{B}]\!]_B \approx [\![\mathcal{U}]\!]_B \approx [\![\mathcal{U}']\!]_S$.*

We remark that when restricted to languages, this corresponds to the result from [9] since (i) if there are non-deterministic Büchi automata $\mathcal{U}$ and $\mathcal{U}'$ (without counters) recognizing a language and its complement, respectively, then $[\![\mathcal{U}]\!]_B = [\![\mathcal{U}']\!]_S$ and (ii) quasi-weak and weak automata coincide when the automata have no counters.

5 Conclusion

We have introduced quasi-weak cost automata as a variant of weak automata which uses the counters to bound the number of alternations between accepting and rejecting states. We have shown quasi-weak cost automata are strictly more expressive than weak cost automata over infinite trees. Moreover, it is the quasi-weak class of automata, rather than the more traditional weak cost automata, which admits a Rabin-style characterization with non-deterministic B-Büchi and S-Büchi automata. The question of a characterization for weak cost automata over infinite trees remains open (it would likely involve some further restrictions on the actions of the counters in the non-deterministic B-Büchi and S-Büchi automata).

Combined with results from [13], our Rabin-style characterization of quasi-weak automata implies the decidability of $f \preccurlyeq g$ and $f \approx g$ when f, g are defined by quasi-weak B-automata. Consequently, this work extends the class of cost functions over infinite trees for which $\approx$ is known to be decidable. In analogy to the reduction in [5], we believe it may be possible to apply this result in order to obtain a decision procedure which determines whether or not a given language of infinite trees is recognizable by a weak automaton. De-

ciding $\preccurlyeq$ and $\approx$ for all regular cost functions over infinite trees remains a challenging open problem which would imply (by [5]) the decidability of the parity-index problem.

Finally, it was known from [13] that weak cost automata and cost WMSO are equivalent. The logic side of quasi-weak cost automata remains to be explored in future work.

Acknowledgments

We are grateful to Thomas Colcombet for having made this joint work possible, and for many helpful discussions.

References

[1] André Arnold and Damian Niwinski. Continuous separation of game languages. *Fundam. Inform.*, 81(1-3):19–28, 2007.

[2] Mikolaj Bojanczyk and Thomas Colcombet. Bounds in ω-regularity. In *LICS*, pages 285–296. IEEE Computer Society, 2006.

[3] Thomas Colcombet. The Theory of Stabilisation Monoids and Regular Cost Functions. In *ICALP (2)*, volume 5556 of *LNCS*, pages 139–150. Springer, 2009.

[4] Thomas Colcombet and Christof Löding. The nesting-depth of disjunctive mu-calculus. In Michael Kaminski and Simone Martini, editors, *CSL*, volume 5213 of *LNCS*, pages 416–430. Springer, 2008.

[5] Thomas Colcombet and Christof Löding. The non-deterministic Mostowski hierarchy and distance-parity automata. In Luca Aceto, Ivan Damgard, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *LNCS*, pages 398–409. Springer, 2008.

[6] Thomas Colcombet and Christof Löding. Regular cost functions over finite trees. In *LICS*, pages 70–79. IEEE Computer Society, 2010.

[7] Kosaburo Hashiguchi. Limitedness theorem on finite automata with distance functions. *J. Comput. Syst. Sci.*, 24(2):233–244, 1982.

[8] Daniel Kirsten. Distance desert automata and the star height problem. *RAIRO - Theoretical Informatics and Applications*, 39(3):455–509, 2005.

[9] Orna Kupferman and Moshe Y. Vardi. The weakness of self-complementation. In Christoph Meinel and Sophie Tison, editors, *STACS*, volume 1563 of *LNCS*, pages 455–466. Springer, 1999.

[10] David E. Muller, Ahmed Saoudi, and Paul E. Schupp. Alternating automata. The weak monadic theory of the tree, and its complexity. In Laurent Kott, editor, *ICALP*, volume 226 of *LNCS*, pages 275–283. Springer, 1986.

[11] Damian Niwinski and Igor Walukiewicz. Deciding nondeterministic hierarchy of deterministic tree automata. *Electr. Notes Theor. Comput. Sci.*, 123:195–208, 2005.

[12] Michael O. Rabin. Weakly definable relations and special automata. In *Mathematical Logic and Foundations of Set Theory (Proc. Internat. Colloq., Jerusalem, 1968)*, pages 1–23. North-Holland, Amsterdam, 1970.

[13] Michael Vanden Boom. Weak cost monadic logic over infinite trees. Accepted to *MFCS*, 2011. Long version at `www.cs.ox.ac.uk/people/michael.vandenboom/`.

Appendix

A Quasi-Weak Automata

A.1 Proof of Proposition 2

We want to show a structural characterization for quasi-weak B-automata. We will call the following property of a B-Büchi automaton the *cycle condition* : in any reachable cycle containing both accepting and rejecting states, there is some counter which is incremented but not reset. Let $\mathcal{A} = \langle Q, \mathbb{A}, q_0, F, \Gamma, \delta \rangle$ be a B-Büchi automaton, and let $k := |\Gamma|$.

Quasi-weak $\implies$ cycle condition

We assume $\mathcal{A}$ does not satisfy the cycle condition, i.e. $\mathcal{A}$ contains a reachable cycle c in the transition function, with states from F and its complement, and such that for all $\gamma \in \Gamma$, if there is an `ic` for γ in c, then there is a reset for γ in c.

Since c is reachable, there exists an input tree t and a play π of $\mathcal{A}$ on t which reaches c within a finite prefix u, and then repeats c forever : $\pi = u(c)^\omega$. The play π is accepting, but does infinitely many alternations. Moreover, its value is bounded by $val_B(uc)$, since c performs a reset for any increment. We can conclude that $\mathcal{A}$ cannot be quasi-weak.

By contraposition, any quasi-weak automaton satisfies the cycle condition.

Cycle condition $\implies$ quasi-weak

We assume $\mathcal{A}$ satisfies the cycle condition. Let π be an accepting play of $\mathcal{A}$ of finite value n. Let m be the number of alternations between accepting and rejecting states in π. We want to show that if m is sufficiently high, then $val(\pi) > n$, which would be a contradiction.

We will use the Ramsey theorem in order to define a large number R, depending only of $\mathcal{A}$ and n. Let R be the bound given by the Ramsey theorem, ensuring that if a graph G has size R and has edges coloured with $2^k - 1$ colours, it contains a one-colour clique of size $n + 2$. Notice that R only depends on (k, n). It means that if k is fixed, there is a correction function α_R such that we can take $R = \alpha_R(n)$.

Assume m is strictly greater than $2|Q|R$. We can find states $(q_i)_{1 \leq i \leq m}$ such that π visits q_1 to q_m in this order, and moreover all q_i with i even are accepting and all q_i with i odd are rejecting. Let u be the finite infix of π, from q_1 to q_m. For all i, we call x_i the position of q_i in u.

Then there exists a set $I \subset [1, m]$ and a state $q \in Q$ such that $|I| \geq R$, and for all $i \in I$, $q_i = q$. We get that u contains $|I| - 1$ consecutives cycles from q to q, each one with both accepting and rejecting states in it. By the cycle condition, each of these cycles (and any concatenation of several of them) must increment a counter without resetting it.

Consider the complete graph G with set of vertices $\{x_i : i \in I\}$, of size at least R. We define the set of colours $K = \{A \subseteq \Gamma : A \neq \emptyset\}$. We colour edges of G in the following way : for any $i < j$ in I, the colour of the edge between x_i and x_j is A, where A is the set of counters that are incremented but not reset in the path from x_i to x_j in u. The cycle condition ensures that $A \neq \emptyset$.

By choice of R, and since $|K| = 2^k - 1$, there is a clique C of size $n+2$ in G, entirely coloured by some $A \in K$. Let $\gamma \in A$. We can write $C = \{x_{i_1}, \dots, x_{i_{n+2}}\}$, with $i_1 < \cdots < i_{n+2}$. For all $j \in [1, n+1]$, there is an increment of γ and no reset of γ in u, between x_{i_j} and $x_{i_{j+1}}$. It means that between x_{i_1} and $x_{i_{n+2}}$, γ is incremented $n+1$ times and never reset. This implies $val(\pi) \geq n+1$, which is absurd.

Assuming $m > 2|Q|R$ leads to a contradiction, so we get that $m < 2|Q|R$ for $R = \alpha_R(n)$, with α_R a correction function given by the Ramsey theorem, and depending only on k.

We can conclude that any accepting play of $\mathcal{A}$ of value n does at most $2|Q|\alpha_R(n)$ alternations between accepting and rejecting states, so $\mathcal{A}$ is a quasi-weak automaton.

A.2 Proof of Proposition 3

We give an explicit cost function f that witnesses the proposition, over alphabet $\mathbb{A} = \{\vee, \wedge\} \times \{e, a, b\}$. We will call $\pi_1 : \mathbb{A} \to \{\vee, \wedge\}$ and $\pi_2 : \mathbb{A} \to \{e, a, b\}$ the projections of $\mathbb{A}$ onto its components.

If t is a $\mathbb{A}$-labelled tree, we will say that a subset C_t of $pos(t)$ is a *choice tree* for t if :

- $\epsilon \in C_t$;
- for all $x \in C_t$, if $\pi_1(t(x)) = \vee$ then C_t contains either the left child or the right child of x;
- for all $x \in C_t$, if $\pi_1(t(x)) = \wedge$ then C_t contains both children of x;
- for all $x \in C_t$ such that $\pi_2(t(x)) = a$, then all paths of C_t starting in x contain a b-labelled position.

Moreover, if C_t is a choice tree of t, we define its value $val(C_t)$ as the maximum number of a's on some path (according to the labelling t).

We now define the desired cost function $f(t) = \inf\{val(C_t) : C_t \text{ is a choice tree of } t\}$. In particular, if there is no choice tree of t, then $f(t) = \infty$.

It is easy to show that f is recognized by a non-deterministic quasi-weak B-automaton : it has two states, and guesses a choice tree while counting the number of a's. The accepting state corresponds to the case where any a has been followed by a b, and a new a makes the automaton go to the rejecting state, until the next b is seen (such a b exists on all paths of the choice tree). So the number of alternations for choice tree C_t is bounded by $2val(C_t)$. Since $val(C_t)$ is the value of the run corresponding to the choice of C_t, this describes a non-deterministic quasi-weak automaton.

Now assume for the sake of contradiction that f is recognized by a weak B-automaton $\mathcal{A}$, up to some correction function α. The states of $\mathcal{A}$ can be partitioned into $Q = Q_0 \uplus Q_1 \uplus \cdots \uplus Q_m$, such that any cycle stabilizes in some Q_i and there are no transitions $Q_i \to Q_j$ with $i < j$. Let $N = |Q|$.

We inductively build a family of $\mathbb{A}$-labelled trees $(t_n)_{n \in \mathbb{N}}$ (positions are labelled by $\{0,1\}^*$):

The tree t_0 is entirely labelled by $(\wedge, e)$. If t_{n-1} is built, we build t_n in the following way:

- $t_n(0^*) = (\wedge, e)$
- $t_n(0^*1) = (\vee, a)$
- $t_n(0^*11^+) = (\vee, e)$
- $t_n(0^*1^N1^+0) = (\wedge, b)$

- Subtrees rooted in nodes of $0^*1^N1^+00^+$ are t_{n-1}
- Other subtrees (to complete the binary tree) are t_0

We have $f(t_n) = n$ for all $n \in \mathbb{N}$. The principle behind the definition of these t_n's is that a B-automaton computing f must do $\Theta(n)$ alternations while processing on input t_n. This is trivial for $n = 0$.

Fix $n \in \mathbb{N}$, and consider a strategy for Eve witnessing value $\alpha(n)$ on t_n. Every play consistent with this strategy must stabilize in an accepting partition. In particular, on the branch 0^ω, the play must stabilize in some Q_i at some node 0^d. Since the label is $\wedge$, the subtree rooted in 0^d1 needs to be also accepting.

Eve is forced to reach node 0^d1^{N+1} in order to witness an accepting choice tree for t_n. By choice of N, there must be a cycle between 0^d and 0^d1^{N+1}.

Assume Adam can enforce an increment without reset during this cycle. Then consider the infinite sequence of trees generated by repeating the cycle a finite (but increasing) number of times. On this sequence, the value of f is unchanged, but Adam has a family of strategies witnessing unbounded value for $\mathcal{A}$. It is absurd so we can consider that the stategy of Eve enforces a reset for any increment in the cycle.

Now if this cycle is accepting, repeating it infinitely many times would create a tree accepted by $\mathcal{A}$ with a bounded cost, but with value ∞ for f, since the a at position 0^d1 will never be followed by a b on any path. Since $\mathcal{A}$ recognizes f, we get a contradiction, so this cycle has to be rejecting, say in partition Q_j with $j < i$. Then Eve can read a b by choosing any node $0^d1^N1^+0$, and this eventually goes back to an accepting state on some node of $0^d1^N1^+0^+$ (otherwise a partial version of t_n would not be accepted by $\mathcal{A}$, but would have value 1 for f). Let Q_k be the partition of this accepting state, with $k < j < i$.

We then need to accept t_{n-1} from partition Q_k, and the same reasoning can be applied again to show that we must have another two alternations. By induction, we can conclude that a weak B-automaton computing f needs at least $2n + 1$ partitions, for any $n \in \mathbb{N}$. This is absurd since $\mathcal{A}$ must have a fixed number of partitions, so f cannot be computed by a weak B-automaton.

B BS-Equivalence and History-Determinism

As described in Section 3, we utilize non-deterministic automata with one or both counter types, on infinite words and on infinite trees. We will write $\mathbb{C}$ to denote the alphabet of counter actions $\mathbb{B}^{\Gamma_B}$, $\mathbb{S}^{\Gamma_S}$ or $\mathbb{B}^{\Gamma_B} \times \mathbb{S}^{\Gamma_S}$ depending on the type(s) of counters.

In addition to the B- and S-semantic introduced earlier, if $\mathcal{A}$ is a non-deterministic BS-automaton, we define the *S-semantic relative to the B-value* $[\![\mathcal{A}]\!]_S^B : \mathbb{N} \to \mathbb{A}^\omega \to \mathbb{N}_\infty$, which seeks to maximize the value over S-accepting runs which also have some bounded B-value:

$$[\![\mathcal{A}]\!]_S^B(m)(u) := \sup \{val_S(R) : R \text{ is an } S\text{-accepting run of } \mathcal{A} \text{ on } u \text{ with } val_B(R) \leq m\}.$$

The *BS-equivalence* can be rewritten in terms of this S-semantic relative to the B-value: $\mathcal{A} \cong_\alpha \mathcal{A}'$ if $[\![\mathcal{A}]\!]_B \approx_\alpha [\![\mathcal{A}']\!]_B$, and for all $m \in \mathbb{N}$ there exists β_m such that $[\![\mathcal{A}]\!]_S^B(m) \preccurlyeq_{\beta_m} [\![\mathcal{A}']\!]_S^B(\alpha(m))$ and $[\![\mathcal{A}']\!]_S^B(m) \preccurlyeq_{\beta_m} [\![\mathcal{A}]\!]_S^B(\alpha(m))$, where $\preccurlyeq$ and $\approx$ are the usual cost function comparison and equivalence. We say that $\mathcal{A}$ and $\mathcal{A}'$ are *BS-equivalent*, written $\mathcal{A} \cong \mathcal{A}'$, if there exists α such that $\mathcal{A} \cong_\alpha \mathcal{A}'$.

The idea is that the B-value is preserved between $\mathcal{A}$ and $\mathcal{A}'$ as usual, and the S-value relative to the B-value is also preserved (after adjusting for any differences in the bound on the B-value).

All of the definitions are the same for non-deterministic BS-automata on infinite trees (now $[\![\mathcal{A}]\!]_S^B$ is a function from $\mathbb{N} \to \mathcal{T}_{\mathbb{A}} \to \mathbb{N}_\infty$).

B.1 History-Determinism

Although cost automata on infinite words cannot always be made deterministic, they can always be made "history-deterministic", a weaker notion introduced in [3] which still retains some nice properties (we refer the interested reader to [3, 6]). Unlike deterministic automata which have a single transition fixed by the current input letter and current state, history-deterministic automata require the entire history of the input and the current state (as well as some 'goal' value for the automaton) in order to uniquely define a transition.

Let $\mathcal{A}$ be a non-deterministic B-automaton or S-automaton with transitions $\Delta : Q \times \mathbb{A} \to \mathcal{P}(\mathbb{C} \times Q)$ where $\mathcal{P}(\mathbb{C} \times Q)$ is the set of subsets of $(\mathbb{C} \times Q)$, a subset being viewed as a disjunction of elements.

A *translation strategy* for $\mathcal{A}$ is a function $\delta : \mathbb{A}^* \times Q \times \mathbb{A} \to \mathbb{C} \times Q$ such that for all $u \in \mathbb{A}^*$ and $(q, a) \in Q \times \mathbb{A}$, $\delta(u, q, a) \in \Delta(q, a)$.

A run R of $\mathcal{A}$ is driven by δ if, after reading u, when the current configuration is (q, a), the next transition in R is $\delta(u, q, a)$.

If $\mathcal{A}$ is a B-automaton (resp. S-automaton), then we say that $\mathcal{A}$ is *history-deterministic* if there is a family of strategies $\delta = (\delta_n)_{n \in \mathbb{N}_\infty}$ such that for all $n \in \mathbb{N}$ and all $u \in \mathbb{A}^\omega$, if R_n is the run of $\mathcal{A}$ on u driven by δ_n, then $val_B(R_n) \leq n$ (resp. $val_S(R_n) > n$) and moreover, if $\mathcal{A}^\delta$ is the automaton $\mathcal{A}$ restricted to the runs R_n, we have $\mathcal{A}^\delta \approx \mathcal{A}$.

The reason history-deterministic automata are useful is that they compose well with alternating automata and games. One can think of $\mathcal{A}$ as reading the output (counter actions, Büchi states, etc.) from a play in the original game, and outputting different actions. The idea is that if there is a history-deterministic automaton $\mathcal{A}$ which computes the valuation used for plays in some game $\mathcal{G}$, then the game $\mathcal{A} \circ \mathcal{G}$ which makes explicit the state of this automaton on all plays still computes the same value (up to $\approx$). For an arbitrary non-deterministic automaton this would not necessarily be possible, because the automaton could disagree about moves on input which share some common prefix. Because $\mathcal{A}$ may use a different valuation itself, this allows us to change the valuation used in the game while preserving the semantic. This is captured by the following lemma which is proven in [13, Lemma 4 in long version].

LEMMA 11. *Let $\mathcal{A}$ be a history-deterministic non-deterministic cost automaton (over infinite words) and let $\mathcal{G}$ be a cost game with valuation f. If $[\![\mathcal{A}]\!] \approx f$, then $val(\mathcal{G}) \approx val(\mathcal{A} \circ \mathcal{G})$.*

Recall that the BS-semantic is not symmetric in B and S : it fixes a B-value and looks at the evolution of S-values among different runs that respect the B-constraint. Therefore, we define a notion of history-determinism that will be more related to the S-side of the automaton. Basically, we want to guarantee that (i) all runs of the automaton preserve the B-value (up to $\approx$) and (ii) all runs driven by translation strategies (δ_n) preserve the S-value (up to $\approx$).

Formally, we say that a BS-automaton $\mathcal{A}$ is *history-deterministic* if (i) there is some α such that for all $u \in \mathbb{A}^\omega$, if R and R' are runs of $\mathcal{A}$ on u, then $val_B(R) \approx_\alpha val_B(R')$ and (ii) there is a family of strategies $\delta = (\delta_n)_{n\in\mathbb{N}_\infty}$ such that for all $n \in \mathbb{N}$ and all $u \in \mathbb{A}^\omega$, if R_n is the run of $\mathcal{A}$ on u driven by δ_n, then $val_S(R_n) = n$ or $val_S(R_n) = \infty$ (but R_n may not be S-accepting); and moreover, if $\mathcal{A}^\delta$ is the automaton $\mathcal{A}$ restricted to the runs R_n, we have $[\![\mathcal{A}^\delta]\!]_S \approx [\![\mathcal{A}]\!]_S$.

Remark that this implies that $\mathcal{A}^\delta \approxeq \mathcal{A}$.

We can take the family δ to be *monotonic*, i.e. if R_n is accepting, then all R_k for $k < n$ are also accepting, and moreover, for any word, R_0 is accepting.

We get a new version of Lemma 11.

LEMMA 12. *Let $\mathcal{A}$ be a history-deterministic non-deterministic BS-automaton (over infinite words) and let $\mathcal{G}$ be a BS-game with valuation f. If $\mathcal{A} \approxeq f$, then $val(\mathcal{G}) \approxeq val(\mathcal{A} \circ \mathcal{G})$.*

C Simulation Theorem

We aim to prove the simulation result (one direction of Theorem 5). We start by recalling a classical lemma, and proving a similar result in the cost setting which will be useful in the simulation proofs. Recall that a regular language L of infinite words is recognizable by a deterministic Büchi automaton iff $L = \lim U$ for some regular language U (where $\lim U = \{u : u(0)\dots u(i) \in U \text{ for infinitely many } i \in \mathbb{N}\}$). We can generalize part of this result to the cost setting as follows.

LEMMA 13. *Let g be a regular cost function over finite words and let*

$$f(u) = \inf\{n : \exists \textit{ infinitely many prefixes } v \textit{ of } u \textit{ such that } g(v) \leq n\},$$
$$f'(u) = \sup\{n : \exists \textit{ infinitely many prefixes } v \textit{ of } u \textit{ such that } g(v) \geq n\}$$

be cost functions over infinite words. Then f (respectively, f') is recognizable by a history-deterministic B-Büchi (respectively, S-Büchi) automaton.

PROOF. We can assume that there is a α_{hd}-history-deterministic B-automaton $\mathcal{A}_{\text{fin}}$ (over finite words) with a family δ of translation strategies such that for all u we have

$$f(u) \approx_\alpha \inf\{n : \exists \text{ infinitely many prefixes } v \text{ of } u \text{ such that } [\![\mathcal{A}_{\text{fin}}]\!](v) \leq n\}$$

which is possible by [3, Theorem 1]. We claim that the same automaton viewed as a B-Büchi automaton and denoted by $\mathcal{A}$ is equivalent to f and is α_{hd}-history-deterministic (witnessed by the same family δ of translation strategies). It suffices to show that $[\![\mathcal{A}]\!]^\delta \preccurlyeq f \preccurlyeq [\![\mathcal{A}]\!]$.

$[\![\mathcal{A}]\!]^\delta \preccurlyeq f$. Assume f is bounded by N on some set U of input words. Let $\beta := \alpha \circ \alpha_{\text{hd}}$. Then for every $u \in U$, there must be some infinite set of indices I such that $[\![\mathcal{A}_{\text{fin}}]\!]^\delta$ is bounded by $\beta(N)$ on prefixes $u(0)\dots u(i)$ for all $i \in I$. This means that using the translation strategy $\delta_{\beta(N)}$ to drive the automaton $\mathcal{A}_{\text{fin}}$ on $u(0)\dots u(i)$ results in an accepting run bounded by $\beta(N)$. Moreover, because $\delta_{\beta(N)}$ deterministically specifies how to construct the run, the runs driven by $\delta_{\beta(N)}$ on prefixes $u(0)\dots u(i)$ and $u(0)\dots u(i')$ are identical on any shared prefix. Thus, this same translation strategy $\delta_{\beta(N)}$ can be used to drive an infinite run of $\mathcal{A}$ on u which witnesses infinitely many accepting states (at each $i \in I$) and has value bounded by $\beta(N)$.

$f \preccurlyeq [\![\mathcal{A}]\!]$. If $[\![\mathcal{A}]\!]$ is bounded by N on some set U, then for any $u \in U$, there is a run of $\mathcal{A}$ on u with value bounded by N and with infinitely many accepting states at positions indexed by some infinite set I. Thus, for all $i \in I$, there is a run of $\mathcal{A}_{\text{fin}}$ on $u(0)\dots u(i)$ which ends in an accepting state and has valued bounded by N. Hence, we have that $\inf\{n : \exists \text{ infinitely many prefixes } v \text{ of } u \text{ such that } [\![\mathcal{A}_{\text{fin}}]\!](v) \leq n\} \leq N$, so $f(u) \leq \alpha(N)$.

The proof for f' is similar. We start with an α'_{hd}-history-deterministic S-automaton $\mathcal{A}'_{\text{fin}}$ (over finite words) with translation strategies δ' such that

$$f'(u) \approx_{\alpha'} \sup\left\{n : \exists \text{ infinitely many prefixes } v \text{ of } u \text{ such that } [\![\mathcal{A}'_{\text{fin}}]\!](v) \geq n\right\}.$$

We prove that $[\![\mathcal{A}']\!] \preccurlyeq f' \preccurlyeq [\![\mathcal{A}']\!]^{\delta'}$ where $\mathcal{A}'$ is $\mathcal{A}'_{\text{fin}}$ viewed as an S-Büchi automaton.

$f' \preccurlyeq [\![\mathcal{A}']\!]^{\delta'}$. Assume that $[\![\mathcal{A}']\!]^{\delta'}$ is bounded by N on some U but f' is unbounded on it. Then there is $u \in U$ such that $f'(u) > \beta'(N)$ where $\beta' := \alpha' \circ \alpha'_{\text{hd}}$, but $[\![\mathcal{A}']\!]^{\delta'}(u) \leq N$. But $f'(u) > \beta'(N)$ implies $\sup\left\{n : \exists \text{ infinitely many prefixes } v \text{ of } u \text{ such that } [\![\mathcal{A}'_{\text{fin}}]\!](v) \geq n\right\} > \alpha'_{\text{hd}}(N)$, and $\sup\left\{n : \exists \text{ infinitely many prefixes } v \text{ of } u \text{ such that } [\![\mathcal{A}'_{\text{fin}}]\!]^{\delta'}(v) \geq n\right\} > N$. This means that there are infinitely many prefixes of u such that the runs of $\mathcal{A}'_{\text{fin}}$ driven by δ'_{N+1} are accepting with value greater than N. Thus δ'_{N+1} on u witnesses an accepting run of value greater than N, contradicting $[\![\mathcal{A}']\!]^{\delta'}(u) \leq N$.

$[\![\mathcal{A}']\!] \preccurlyeq f'$. Assume f' is bounded by N on some set U. Then for any $u \in U$, we have $\sup\left\{n : \exists \text{ infinitely many prefixes } v \text{ of } u \text{ such that } [\![\mathcal{A}'_{\text{fin}}]\!](v) \geq n\right\}$ must be bounded by $\alpha'(N)$. Assume by contradiction that there is some $u \in U$ such that $[\![\mathcal{A}']\!](u) > \alpha'(N)$. Then there is an accepting run of $\mathcal{A}'$ on u with checked values greater than $\alpha'(N)$. If the accepting states are at positions indexed by some infinite set I, then for all $i \in I$, $\mathcal{A}'_{\text{fin}}$ acting on the prefixes $u(0)\dots u(i)$ accepts with value greater than $\alpha'(N)$. But this contradicts the fact that $\sup\left\{n : \exists \text{ infinitely many prefixes } v \text{ of } u \text{ such that } [\![\mathcal{A}'_{\text{fin}}]\!](v) \geq n\right\} \leq \alpha'(N)$. ∎

C.1 Proof of Lemma 6

We aim to simulate an alternating B-Büchi automaton $\mathcal{B}$ with a non-deterministic B-Büchi automaton $\mathcal{U}$.

Given an input tree t and a finite-memory strategy σ for Eve in the B-Büchi game $(\mathcal{B}, t)$, we consider the tree t_σ which is annotated with this strategy. In other words, t_σ uses the extended alphabet $\mathbb{A}' := \mathbb{A} \times \mathcal{P}((Q \times S) \times \mathbb{B} \times (Q \times S) \times [0,1])$ where Q is the set of states from $\mathcal{B}$, S is the additional memory needed for the strategy, $\mathbb{C}$ is the set of actions in a B-Büchi game, and $[0,1]$ are the possible directions in a binary tree. Moreover, $t_\sigma(x) := (t(x), S_\sigma(x))$ where $S_\sigma(x)$ is

$$\{((q,s),c,(q,s'),k) : (c,(q',xk)) \text{ is possible from } (q,x) \text{ via } \sigma \text{ and } s,s' \in S\}.$$

Let $\tau = k_0k_1\ldots \in [0,1]^\omega$ be a path in t. Given t_σ and τ, let $w^\tau_\sigma := (a_0,k_0)(a_1,k_1)\ldots$ such that $a_j = t_\sigma(k_0k_1\ldots k_j) \in \mathbb{A}'$, so w^τ_σ is the word that describes the plays compatible with σ which stay on τ.

LEMMA 14. *There is a history-deterministic B-Büchi automaton $\mathcal{D}_{\max}$ which recognizes the function max-play$(w_\sigma^\tau) = \sup\{val(\pi) : \pi$ is compatible with σ and stays on $\tau\}$.*

PROOF. Given $w := w_\sigma^\tau$, *max-play* can be rewritten as $\max\{val_{\text{Büchi}}(w), val_{\text{counters}}(w)\}$ where $val_{\text{Büchi}}(w)$ is ∞ if there is some play π described by w which does not satisfy the Büchi acceptance condition and 0 otherwise, and $val_{\text{counters}}(w)$ is the maximum counter value achieved on any play π described by w. It is easy to see that history-deterministic B-Büchi automata are closed under max. So it suffices to show $val_{\text{Büchi}}$ and val_{counters} are recognizable by history-deterministic B-Büchi automata.

We first describe informally a deterministic Büchi automaton $\mathcal{D}$ recognizing $val_{\text{Büchi}}$ (this automaton is based on a construction from [10]). The idea is that the state of $\mathcal{D}$ includes a "testing set" $E \subseteq Q$ (representing a subset of possible states the automaton could be in based on the word w that is being read). We write E_i for the testing set at position i in the word. The set E_0 is the set of rejecting states described in $w(0)$. If $E_i = \varnothing$, then E_{i+1} is the set of rejecting states described in $w(i+1)$. Otherwise, the testing set E_{i+1} is updated to include all rejecting states q_{i+1} reachable from rejecting states $q_i \in E_i$ by the moves described in $w(i)$ (note that rejecting states reachable from accepting states in $w(i)$ are not added). The Büchi acceptance condition is used to ensure that the testing set is $\varnothing$ infinitely often, i.e. there is no play described in w which stabilizes in rejecting states.

Next, we show that there is a history-deterministic B-Büchi automaton $\mathcal{E}$ recognizing val_{counters}. Since val_{counters} can ignore the Büchi condition, we can actually obtain the maximum value of counters by looking at larger and larger prefixes of the plays. That is, $val_{\text{counters}} = \inf\{n : \exists$ infinitely many prefixes v of w s.t. $val^{\text{fin}}_{\text{counters}}(v) \leq n\}$, where we write $val^{\text{fin}}_{\text{counters}}(v)$ for the maximum counter value achieved on the partial (finite) plays described by v. By Lemma 13, if $val^{\text{fin}}_{\text{counters}}$ is a regular cost function on finite words, then val_{counters} is recognizable by a history-deterministic B-Büchi automaton.

Thus, it remains to show that $val^{\text{fin}}_{\text{counters}}$ is a regular cost function. Given a partial play π, there is an S-automaton which recognizes $val_B(\pi)$ (by [6, Lemma 4]). We can then construct a new S-automaton which when reading a prefix of w, non-deterministically selects a partial play described by it, checks that it is a valid partial play in the game, and then computes its val_B value using the previous automaton. Given some prefix v of w, this automaton recognizes the maximum of val_B over the partial plays described in v, which is exactly $val^{\text{fin}}_{\text{counters}}$. ∎

Now we construct a non-deterministic B-Büchi automaton $\mathcal{U}$ which on input t non-deterministically selects annotations for t, checks that these annotations correspond to an actual finite-memory strategy σ in the game $(\mathcal{B}, t)$ (and rejects if not), and then simulates $\mathcal{D}_{\max}$ on each branch of t_σ (this is possible since $\mathcal{D}_{\max}$ is history-deterministic by Lemma 14).

Recall that the value of a strategy in a B-Büchi game is the maximum value of the plays. Thus, running $\mathcal{D}_{\max}$ on each branch of t_σ yields a value $\approx$-equivalent to $val(\sigma)$.

Moreover, since finite-memory strategies suffice in B-Büchi games by [13, Theorem 4], the overall value (up to $\approx$) of $(\mathcal{B}, t)$ is the infimum over the values of all finite-memory strategies for Eve. But non-determinism in the B-Büchi automaton $\mathcal{U}$ resolves into taking an infimum. Thus, $\mathcal{U}$ finds the minimum value across all finite-memory strategies in $(\mathcal{B}, t)$, so

$[\![\mathcal{B}]\!] \approx [\![\mathcal{U}]\!]$.

Note that we have shown that any alternating B-Büchi automaton can be simulated by a non-deterministic B-Büchi automaton (since we did not need the quasi-weak condition anywhere in this section). This is no longer true in the S-case below.

C.2 Proof of Lemma 7

The simulation of quasi-weak B-automata with non-deterministic S-Büchi automata is more technical because we must first introduce more general objectives/cost games which are used in the proofs. The overall method, however, is the same as the B-Büchi case: the non-deterministic automaton will guess a finite-memory strategy in a game related to the original alternating quasi-weak B-automaton, and then run a history-deterministic cost-automaton on each branch in order to compute the value of this strategy (this time it is an S-Büchi automaton which is run on each branch).

Objectives in Cost Games

So far we have only worked with the B-Büchi and S-Büchi valuations, and the games which use these valuations. These can actually be viewed in the context of cost games with more general objectives $O = \langle \mathbb{C}, f, goal \rangle$ where $\mathbb{C}$ specifies the counter actions, $f : \mathbb{C}^\omega \to \mathbb{N}_\infty$ is the valuation for those actions, and $goal \in \{\min, \max\}$ specifies how the player seeks to optimize f in the game. We describe these ideas formally now.

Formally, a *cost game* $\mathcal{G} := \langle V, v_0, \delta, O \rangle$ consists of a set of positions V, an initial position $v_0 \in V$, an objective $O = \langle \mathbb{C}, f, goal \rangle$ for Eve, and a control function $\delta : V \to \mathcal{B}^+(\mathbb{C} \times V)$ (where $\mathcal{B}^+(\mathbb{C} \times V)$ is the set of positive boolean combinations, written as a disjunction of conjunctions of elements from $\mathbb{C} \times V$).

The objective $O = \langle \mathbb{C}, f, goal \rangle$ describes how to assign values to the objects in the game. For a play $\pi = (v_i, c_{i+1}, v_{i+1})_{i \in \mathbb{N}}$, the value is $val(\pi) := f(\pi_{\mathbb{C}})$ where $\pi_{\mathbb{C}} := c_1 c_2 \ldots$. If $goal$ is min, then the value of a strategy σ for Eve is $val(\sigma) := \sup\{val(\pi) : \pi \in \sigma\}$ and the value of the game is $val(\mathcal{G}) := \inf\{val(\sigma) : \sigma \text{ is a strategy for Eve in } \mathcal{G}\}$. In other words, Eve seeks to minimize over all strategies the maximum value of all plays compatible with the strategy. Dually, if $goal$ is max, then $val(\sigma) := \inf\{val(\pi) : \pi \in \sigma\}$ and $val(\mathcal{G}) := \sup\{val(\sigma) : \sigma \text{ is a strategy for Eve in } \mathcal{G}\}$.

The *dual* $\overline{\mathcal{G}}$ of a game $\mathcal{G}$ is obtained by switching disjunctions and conjunctions in the control function and using the dual objective (i.e. replacing min with max, and vice versa). This switches the roles of Adam and Eve. Martin's theorem implies that a cost game and its dual have the same value.

PROPOSITION 15. $val(\mathcal{G}) = val(\overline{\mathcal{G}})$.

Note that in this formulation of cost games all of the information required to determine the value of a play must appear on the transitions. Thus, in a B-Büchi game we view a state as producing an output of priority 1 if it is rejecting and priority 2 if it is accepting (see [13] for more information about these "cost-parity" games). Unless indicated otherwise, we set $\mathbb{C} := \mathbb{B}^\Gamma \times [1, 2]$, the output alphabet from a B-Büchi game with counters Γ.

This means that a B-Büchi game with counters Γ has objective $\langle \mathbb{C}, cost_B^{\Gamma,[1,2]}, \min \rangle$ where $cost_B^{\Gamma,[1,2]}$ assigns value ∞ if the play stabilizes in priority 1 (i.e. it is rejecting according to the Büchi condition), and otherwise assigns $val_B(u)$ (i.e. the maximum checked value of any counter). The game $(\mathcal{A}, t)$ described in Section 2 for $\mathcal{A}$ an alternating B-Büchi automaton and t an input tree can easily be translated into a game of this form.

Recall that we intend on using the non-determinism of the S-automaton to guess a strategy, and the non-determinism in S-automata resolves into taking a supremum. Thus, we need to consider a game which has max as the goal. The natural candidate (and the one used, for instance, in the simulation proof in [6]) is the dual of a B-Büchi game, which we call a *$\overline{B}$-Büchi game* (with objective $\langle \mathbb{C}, cost_B^{\Gamma,[1,2]}, \max \rangle$). We use, as an intermediate object, a related game called a $\overline{B}$-safety game.

$\overline{B}$-Safety Games

Observe that in a quasi-weak $\overline{B}$-game, a play is assigned value ∞ if there are infinitely-many priority 1 (since this means either the play has stabilized in priority 1 and is rejecting, or it has infinitely-many alternations and consequently the B-value is ∞ by the quasi-weak hypothesis). As a result, it is helpful to know where these priority 1 transitions occur.

We now consider a variant of a $\overline{B}$-Büchi game which we call a *$\overline{B}$-safety game*. A $\overline{B}$-safety game has additional signals \$ added in the output. These signals are designed to provide more information about the structure of the game in terms of the Büchi acceptance condition, namely where these priority 1 transitions occur. This additional structure (i) makes it easier to approximate the value of an infinite play based on the values of finite prefixes of the play (which is needed in order to apply Lemma 13), and (ii) makes it easier to prove that finite-memory strategies suffice in quasi-weak $\overline{B}$-Büchi games.

Formally, a $\overline{B}$-safety game has objective $\langle \mathbb{C}', \textit{safety-cost}_{\overline{B}}^{\Gamma,[1,2]}, \max \rangle$ where $\mathbb{C}' := \mathbb{C} \cdot \{\epsilon, \$\}$ and $\textit{safety-cost}_{\overline{B}}^{\Gamma,[1,2]}(u)$ is defined as follows. If there are finitely many \$ in u, then we set $\textit{safety-cost}_{\overline{B}}^{\Gamma,[1,2]}(u) := 0$. Otherwise, u can be split into $u_0\$u_1\$\dots$ such that each u_i contains no \$. If there is some u_i in u such that there is no priority 1 in u_i (but there is priority 1 in each $u_{i'}$ for $i' < i$), then $\textit{safety-cost}_{\overline{B}}^{\Gamma,[1,2]}(u) := val_B(h_{[1,2]}(u_0u_1 \dots u_i))$, where $h_{[1,2]}$ removes the priorities, keeping only counter actions. In that case, the value is the normal B-value on the prefix $u_0u_1 \dots u_i$. Otherwise, $\textit{safety-cost}_{\overline{B}}^{\Gamma,[1,2]}(u) := \infty$. We write $\textit{safety-cost}_{\overline{B}}^{\Gamma}(u)$ for the $\overline{B}$-safety valuation on finite words u which assigns value 0 if u does not end in \$, $val_B(h_{[1,2]}(u_0u_1 \dots u_i))$ if $u = u_0\$u_1\$\dots\$u_j\$$ where each $u_{i'}$ for $i' \leq j$ contains no \$ and u_i for $i \leq j$ is the first subword which does not contain priority 1, and value ∞ otherwise.

The idea is that it is more difficult to obtain a high value in a quasi-weak $\overline{B}$-safety game compared to a quasi-weak $\overline{B}$-game because of the additional requirements enforced by \$ (e.g., a play with infinitely many priority 1 has value ∞ in a quasi-weak $\overline{B}$-game, but a play with infinitely many signals in a $\overline{B}$-safety game must not only have infinitely many priority 1 but must also witness priority 1 between each signal in order to be assigned value ∞).

We say a $\overline{B}$-safety game $\mathcal{G}'$ is *based on* a $\overline{B}$-Büchi game $\mathcal{G}$ with an acyclic game graph if the arena and transitions are identical except for the fact that signals \$ have been added to

some edges in the game graph such that either all transitions at some depth have output \$, or none do. For $\overline{B}$-safety games based on quasi-weak $\overline{B}$-games, there is a nice lemma about the relationship between a quasi-weak $\overline{B}$-game and the $\overline{B}$-safety games based on it.

LEMMA 16. *For every quasi-weak $\overline{B}$-game $\mathcal{G}$ with an acyclic game graph, we have $val(\mathcal{G}) = \sup\{val(\mathcal{G}') : \mathcal{G}' \text{ is a } \overline{B}\text{-safety game based on } \mathcal{G}\}$.*

PROOF. Assume a strategy τ for a quasi-weak $\overline{B}$-game $\mathcal{G} = \langle V, v_0, \delta, \langle \mathbb{C}, cost_B^{\Gamma,[1,2]}, \max\rangle\rangle$ with an acyclic game graph witnesses $val(\mathcal{G}) = val(\tau) = n$. Let T be the corresponding strategy tree or run tree (i.e. the tree of all plays compatible with strategy τ).

We define inductively a strictly increasing sequence of depths $(d_i)_{i\in\mathbb{N}}$ where $d_0 = 0$ and d_{i+1} is the least d such that all paths in T either (a) have counters which witness value n before depth d or (b) have at least one transition labelled with priority 1 between depths d_i and d. This is well-defined: assume by contradiction that there is some position $s \in T$ from which there is no bound on the depth at which (a) or (b) are satisfied. Because of the finite branching inherent in these games, König's Lemma would imply that there is a path from s on which neither (a) nor (b) are satisfied, i.e. a path in T through s with only finitely many priority 1 transitions (and no priority 1 transitions after s) and counter values less than n, so this path has value less n. But this means $val(\tau) < n$, contradicting our initial assumption.

Next we transform $\mathcal{G}$ into a $\overline{B}$-safety game $\mathcal{G}_\tau = \langle V, v_0, \delta', O'\rangle$ by updating the output to produce the appropriate signals at depths $(d_i)_{i\in\mathbb{N}}$ and setting the objective to $O' = \langle \mathbb{C}\cdot\{\epsilon,\$\}, \textit{safety-cost}_{\overline{B}}^{\Gamma,[1,2]}, \max\rangle$. Playing according to τ in $\mathcal{G}_\tau$ (and adding \$ at the depths $(d_i)_{i\in\mathbb{N}}$) witnesses the fact that $\sup\{val(\mathcal{G}') : \mathcal{G}' \text{ is a } \overline{B}\text{-safety game based on } \mathcal{G}\} \geq val(\mathcal{G})$.

Now assume by contradiction that there is a strategy σ in a $\overline{B}$-safety game $\mathcal{G}'$ based on $\mathcal{G}$ such that $val(\sigma) > val(\mathcal{G})$. This means that on each play $\pi \in \sigma$, either there are infinitely many priority 1 in π (with at least one such transition between each signal \$) or the counters witness a value exceeding n. In either case, $cost_B^{\Gamma,[1,2]}(h_\$(\pi)) > n$ where $h_\$$ removes the signal output. (Note that this is not necessarily true when $\mathcal{G}'$ is based on an arbitrary $\overline{B}$-Büchi game, but is true when it is based on a quasi-weak $\overline{B}$-game $\mathcal{G}$ as assumed here.) But this means that $val(\mathcal{G}) > n$, contradicting the initial assumption. ∎

Finite Memory Strategies

Next, we show that finite-memory strategies suffice in quasi-weak $\overline{B}$-games with acyclic game graphs: that is, there is some correction function α such that if there is a strategy in a quasi-weak $\overline{B}$-game which can guarantee a value of at least $\alpha(n)$, then there is a finite-memory strategy which can guarantee a value of n. We actually do this by translating between different types of games/objectives: quasi-weak $\overline{B}$-games, quasi-weak $\overline{hB}$-games, and $\overline{hB}$-safety games.

Fix an arbitrary strategy τ that witnesses at least cost $(n+1)^k$ for the quasi-weak $\overline{hB}$-game $\mathcal{G} = \langle V, v_0, \delta, O\rangle$ with $O = \langle \mathbb{C}, cost_{hB}^{[1,k],[1,2]}, \max\rangle$ and $\mathbb{C}$ the actions in a hierarchical B-Büchi game. We assume that $\mathcal{G}$ has an acyclic game graph and uses k hierarchical counters. Let T be the corresponding strategy tree (the tree of plays compatible with τ).

We define inductively a strictly increasing sequence of depths $(d_i)_{i\in\mathbb{N}}$ where $d_0 = 0$ and d_{i+1} is the least d such that all paths in T either (a) have counters which witness value $(n+1)^k$ before depth d or (b) have at least one transition labelled with priority 1 between depths d_i and d. This is well-defined (see the proof of Lemma 16).

Next we transform $\mathcal{G}$ into a $\overline{hB}$-safety game $\mathcal{G}_\tau = \langle V, v_0, \delta', O'\rangle$. If v is not at a depth d_i for any $i \in \mathbb{N}$, then $\delta'(v) := \delta(v)$. Otherwise, for v at depth d_i for some $i \in \mathbb{N}$, we need to update the output to produce the appropriate signals described above: $\delta'(v) := \delta(v)[(c\$, v')/(c, v')]$. In order to assign a value to plays, we set the objective to be $O' = \langle \mathbb{C}\cdot\{\epsilon,\$\}, \textit{safety-cost}_{\overline{B}}^{[1,k],[1,2]}, \max\rangle$. Moreover, there is a deterministic $\overline{hB}$-automaton $\mathcal{D}$ over the alphabet $\mathbb{C}' := \mathbb{C}\cdot\{\epsilon,\$\}$ which recognizes $\textit{safety-cost}_{\overline{B}}^{[1,k],[1,2]}$ (it uses the state to remember whether priority 1 has been seen between signals \$, and whether the output is still being analyzed).

This new game must have value at least $(n+1)^k$ (since by adding \$ at appropriate depths, τ can be transformed into a strategy in $\mathcal{G}_\tau$ which witnesses value at least $(n+1)^k$).

LEMMA 17. $\textit{val}(\mathcal{G}_\tau) \geq (n+1)^k$

We now consider the composition $\mathcal{D}\times\mathcal{G}_\tau$, which is a quasi-weak $\overline{hB}$-game since $\mathcal{D}$ translates the game $\mathcal{G}_\tau$ into an $\overline{hB}$ objective (and neither changes the value of cycles of priority 1 and 2, nor introduces any additional cycles with both priorities). The additional structure in this new game can be used to show that positional strategies suffice.

LEMMA 18. *If there is a strategy of value at least $(n+1)^k$ in $\mathcal{D}\times\mathcal{G}_\tau$, then there is a positional strategy of value at least n in $\mathcal{D}\times\mathcal{G}_\tau$.*

PROOF. Fix a strategy τ' witnessing at least cost $(n+1)^k$ in $\mathcal{G}' := \mathcal{D}\times\mathcal{G}_\tau$ and let T' be the corresponding strategy tree. Let $h : S \to V$ be the homomorphism between the set of positions S in T' and the set of positions V in $\mathcal{G}'$.

Using an optimal strategy τ', Eve's choice at a particular $v \in V$ may depend on the history of the play leading to v. Thus, there may be $s, s' \in h^{-1}(v)$ such that the moves possible from s are different than from s'; however, because the game graph is acyclic, s, s', and v must be at the same depth. A positional strategy σ' can be viewed as a mapping from V to S which for each v selects a single element of $h^{-1}(v)$ for Eve to use (regardless of the history). Such a positional strategy may not be able to achieve the optimal value, but the difference between the positional strategy and an arbitrary strategy should be bounded (given by some correction function α). To build this map σ', we use the notion of "signatures".

We define a signature for nodes $s \in S$ as follows: let

$$\textit{sig}(s) := \langle \gamma_{\max}(s), \gamma_1(s), \gamma_2(s), \ldots, \gamma_k(s)\rangle$$

where $\gamma_j(s)$ is the current value of counter j according to the counter actions described on the path from the root of T' to s (up to value $(n+1)^k$), and $\gamma_{\max}(s)$ is $(n+1)^k$ if the path to s has already reached at least value $(n+1)^k$ or $\max\{\gamma_1(s),\ldots,\gamma_k(s)\}$ otherwise. Let $\sigma' : V \to S$ where $\sigma'(v)$ selects the node $s \in h^{-1}(v)$ with the lexicographically-least signature. We extend *sig* to nodes $v \in V$ be setting $\textit{sig}(v) := \textit{sig}(\sigma'(v))$.

It is clear that σ' is a positional strategy (see the earlier explanation). It remains to show that minimizing this signature resulted in a positional strategy which can guarantee value at least n.

Suppose for contradiction that $val(\sigma') < n$, so there is some play $\pi \in \sigma'$ with $val(\pi) = m < n$. Then π must not visit priority 1 infinitely often (otherwise the value of the play would be immediately ∞ by the properties of quasi-weak automata). Thus, there is some least i such that no transitions of priority 1 occur between d_i and d_{i+1}. Let v be the game position at depth d_{i+1} in π. For all $s \in h^{-1}(v)$, there must be no priority 1 between d_i and d_{i+1} (since this is true at v and the state of $\mathcal{D}$ is recorded in the game position). Hence, for all $s \in h^{-1}(v)$, $\gamma_{\max}(s) = (n+1)^k$ so $sig(v) \geq \langle (n+1)^k, 0, \ldots, 0\rangle$. Likewise, we know that the signature at the initial game position v'_0 is $\langle 0, 0, \ldots, 0\rangle$.

We now show that $\langle l + (m+1)^k, \ldots, l + (m+1)^k\rangle$ is an upper bound for the signature possible on a partial play derived from σ' which starts from a position s satisfying $\langle l, 0, \ldots, 0\rangle < sig(s) \leq \langle l, \ldots, l\rangle$ and has value m. Since we are starting from $\langle 0, \ldots, 0\rangle$ and $m < n$, this is enough to yield the desired contradiction.

First, notice that an increment of some counter is necessary (but not sufficient) for a strict increase in a signature $\langle m_0, m_1, m_2, \ldots, m_k\rangle < \langle (n+1)^k, 0, \ldots, 0\rangle$. If counter j is incremented, then the maximum value of the signature at the next position is $\langle \max\{m_j + 1, m_{j+1}, \ldots, m_k\}, 0, \ldots, m_j + 1, m_{j+1}, \ldots, m_k\rangle$. Likewise, if counter j is reset, then the max signature is $\langle \max\{m_{j+1}, \ldots, m_k\}, 0, \ldots, 0, m_{j+1}, \ldots, m_k\rangle$. In all signatures, we have $m_0 \geq \max\{m_1, \ldots, m_k\}$.

If $k = 1$, then by incrementing the counter m times the signature can increase at most to $\langle l + m, l + m\rangle$. The signature cannot be increased further by using resets since a reset results in the signature first decreasing to $\langle 0, 0\rangle$. Thus, the maximum signature is bounded as desired.

Now let $k > 1$. By the inductive hypothesis, the signature $sig_0 = \langle l + (m+1)^{k-1}, \ldots, l + (m+1)^{k-1}, 0\rangle$ is the maximum possible using only counters $\{1, \ldots, k-1\}$ and a partial play of value m. We can assume that any other increments of counters $\{1, \ldots, k-1\}$ would result in some counter exceeding m (otherwise, the signature would not be maximum).

If counter k is reset, then the signature must decrease to $\langle 0, \ldots, 0\rangle$, so that operation is not useful. If counter $j < k$ is reset, then the signature must first decrease to $\langle l + (m+1)^{k-1}, \ldots, 0, l + (m+1)^{k-1}, \ldots, l + (m+1)^{k-1}, 0\rangle$. Again, we can assume that touching counters $\{j+1, \ldots, k-1\}$ would result in a value exceeding m, so only counters $\{1, j\}$ can be touched, which by the inductive hypothesis can only increase the signature back to sig_0.

But $\langle l + (m+1)^{k-1}, 0, \ldots, 0, l + (m+1)^{k-1}\rangle < sig_0$, so σ' could select a node with such a signature. An increment of counter k from this position results in signature $\langle l + (m+1)^{k-1} + 1, 0, \ldots, 0, l + (m+1)^{k-1} + 1\rangle$ (and resets all lower counters). The strategy σ' could now select a lower signature $\langle l + (m+1)^{k-1}, l + (m+1)^{k-1}, \ldots, l + (m+1)^{k-1}\rangle$. We can apply the inductive hypothesis again from here, to get a maximum signature of $sig_1 = \langle l + 2(m+1)^{k-1}, \ldots, l + 2(m+1)^{k-1}, l + (m+1)^{k-1}\rangle$. Because the increment of counter k reset the values, the value of the play according to σ' is still at most m.

Repeating this process, we get a maximum signature of $sig_m = \langle l + (m+1)^k, \ldots, l + (m+1)^k\rangle$. Repeating the process again would result in counter k achieving a value greater than m, so sig_m is the upper bound. ∎

Next, we can use the previous lemma to give a value to the original quasi-weak $\overline{hB}$-game $\mathcal{G}$ which is based on finite-memory strategies in $\overline{hB}$-safety games based on it.

LEMMA 19.
$val(\mathcal{G}) \approx_\alpha \sup\{val(\sigma) : \sigma$ is finite-memory strategy in $\overline{hB}$-safety game $\mathcal{G}'$ based on $\mathcal{G}\}$ with $\alpha(n) = (n+1)^k$.

PROOF. By Lemma 16, one direction is clear: $val(\mathcal{G})$ is at least the supremum of the values of the finite-memory strategies in $\overline{hB}$-safety games based on it. In order to show that $val(\mathcal{G}) \preccurlyeq_\alpha \sup\{val(\sigma) : \sigma$ is a finite-memory strategy in a $\overline{hB}$-safety game $\mathcal{G}'$ based on $\mathcal{G}\}$ where $\alpha(n) = (n+1)^k$, it suffices to show that if $\mathcal{G}$ has value $(n+1)^k$, then there is a $\overline{hB}$-safety game based on it which has a finite-memory strategy of value at least n.

Assume there is a strategy τ witnessing value at least $(n+1)^k$ in $\mathcal{G}$. By Lemma 11 and Lemma 17, we have $val(\mathcal{D} \times \mathcal{G}_\tau) = val(\mathcal{G}_\tau) \geq (n+1)^k$. Since there is a positional strategy σ_τ in $\mathcal{D} \times \mathcal{G}_\tau$ witnessing value at least n by Lemma 18, there is a finite-memory strategy witnessing value at least n in $\mathcal{G}_\tau$ (where the memory depends on $\mathcal{D}$). ∎

A similar proof (which also uses the history-deterministic transducers from B- to hB-games) can be used to show that quasi-weak B-games (and the $\overline{B}$-safety games based on them) admit finite-memory strategies when the underlying game graph is acyclic.

COROLLARY 20. *Finite-memory strategies suffice in quasi-weak $\overline{B}$-games with acyclic game graphs (and $\overline{B}$-safety games based on them).*

Proof of Lemma 7

We use a similar method as in the previous case to construct a non-deterministic S-Büchi automaton $\mathcal{S}$. This time we start by considering trees annotated by a strategies σ in a $\overline{hB}$-safety game based on quasi-weak $\overline{hB}$-game. By fixing a path τ in such a tree, we get a word w_σ^τ as before.

LEMMA 21. *There is a history-deterministic hS-Büchi automaton $\mathcal{D}_{\min}$ which recognizes the function min-play$(w_\sigma^\tau) = \inf\{$safety-cost$_{\overline{B}}^{\Gamma,[1,2]}(\pi) : \pi \in \sigma|_\tau\}$.*

PROOF. Let $w := w_\sigma^\tau$. Then the function *min-play* can be rewritten as *min-play*$(w) = \sup\{n : \exists$ infinitely many prefixes v of w s.t. $g'(v) \geq n\}$ where g' maps a finite prefix of w to the minimum safety-cost over the partial plays described in this prefix (in particular, if I describes an infinite number of positions where \$ occurs in w, then *min-play*$(w) = \sup\{n : \forall i \in I. g'(w(0)\dots w(i)) \geq n\}$). By Lemma 13, it suffices to show that g' is a regular cost function.

Given a partial play π, it is straightforward to construct a B-automaton $\mathcal{C}'$ which recognizes safety-cost$_{\overline{B}}^{\Gamma}(\pi)$ (just copy the output actions from the input word which describes a play, and use the state to track whether priority 1 has been visited between signals in order to determine acceptance). The desired B-automaton recognizing g' non-deterministically selects a partial play in a prefix of w, checks that it is a valid partial play in the $\overline{hB}$-safety game, and simulates $\mathcal{C}'$ on it; this computes the minimum of safety-cost$_{\overline{B}}^{\Gamma}$ over the partial plays described in a prefix of w. ∎

We can assume that $\mathcal{B}$ uses hierarchical counters. The desired automaton $\mathcal{U}'$ is then constructed such that on input t it guesses a $\overline{hB}$-safety game based on $\overline{(\mathcal{B}, t)}$ (the dual of the original game), guesses a finite-memory strategy in this game, and then computes the value of this game by simulating $\mathcal{D}_{\min}$ on each branch. Because $\mathcal{D}_{\min}$ is an S-Büchi automaton, the resulting automaton is a non-deterministic S-Büchi automaton. In an S-automaton, non-determinism amounts to taking a supremum. Hence, the automaton computes the supremum over the values of finite-memory strategies in $\overline{hB}$-safety games based on $\overline{(\mathcal{B}, t)}$, which is $\approx$-equivalent to the value of $\overline{(\mathcal{B}, t)}$ by Lemma 19.

D Transducers for Hierarchical Automata

D.1 B to hB and S to hS

It was already known from [3] that hierarchical counters suffice in cost automata. In [6], the translation from B to hB is shown explicitly, using ideas from the "latest-appearance record" construction. A similar result from S to hS was already known by Colcombet and Löding and we describe the construction here.

Fix some $\Gamma_S = [1, k]$ of counters. Our goal is to build a history-deterministic hS-automaton $\mathcal{A}$ using the same counters Γ_S such that $[\![\mathcal{A}]\!]_S \approx val_S$ and the counter operations are hierarchical. Hierarchical actions are labeled by counter number; if $k = 3$, then action $\mathtt{i}_2$ stands for $(\mathtt{r}, \mathtt{i}, \varepsilon)$.

The state-set Q includes all tuples $(X, \langle \gamma_j, \ldots, \gamma_k \rangle)$ where $X \subseteq \Gamma_S$, $j = |X| + 1$, and $\langle \gamma_j, \ldots, \gamma_k \rangle$ is a permutation of $\Gamma_S \setminus X$, as well as a sink state $\perp$. We write $\langle Y, \gamma_j, \ldots, \gamma_k \rangle$ to denote the permutation which lists the elements of Y in ascending order followed by $\gamma_j, \ldots, \gamma_k$. The initial state is $(\varnothing, \langle \Gamma_S \rangle)$. The only rejecting state is $\perp$.

Now consider a state $(X, \langle \gamma_j, \ldots, \gamma_k \rangle)$ and an input letter $a \in \{\varepsilon, \mathtt{i}, \mathtt{r}, \mathtt{cr}\}^{\Gamma_S}$. We describe the edges in the transition relation Δ. Let $Y = \{\gamma \in X : a(\gamma) \neq \varepsilon\}$. Let $i \in [j, k]$ be the largest index such that $a(\gamma_i) \neq \varepsilon$.

- If such an i does not exist, then $(X, \langle \gamma_j, \ldots, \gamma_k \rangle) \xrightarrow{a:\varepsilon} (X \setminus Y, \langle Y, \gamma_j, \ldots, \gamma_k \rangle)$.
- If $a(\gamma_i) = \mathtt{i}$ then $(X, \langle \gamma_j, \ldots, \gamma_k \rangle) \xrightarrow{a:\mathtt{i}_i} (X \setminus Y, \langle Y, \gamma_j, \ldots, \gamma_k \rangle)$.
- If $a(\gamma_i) = \mathtt{r}$ then $(X, \langle \gamma_j, \ldots, \gamma_k \rangle) \xrightarrow{a:\mathtt{r}_i} (X \setminus Y, \langle Y, \gamma_i, \gamma_j, \ldots, \gamma_{i-1}, \gamma_{i+1}, \ldots, \gamma_k \rangle)$.
- If $a(\gamma_i) = \mathtt{cr}$ then $(X, \langle \gamma_j, \ldots, \gamma_k \rangle) \xrightarrow{a:\varepsilon} \perp$.
- Regardless, $(X, \langle \gamma_j, \ldots, \gamma_k \rangle) \xrightarrow{a:\mathtt{cr}_i} ((X \cup \{\gamma_i\}) \setminus Y, \langle Y, \gamma_j, \ldots, \gamma_{i-1}, \gamma_{i+1}, \ldots, \gamma_k \rangle)$.

Let $w \in \{\varepsilon, \mathtt{i}, \mathtt{r}, \mathtt{cr}\}^{\Gamma_S}$. Let u be an output sequence from $\mathcal{A}$ on w which yields the maximum value (i.e. $[\![\mathcal{A}]\!]_S(w) = val_S(u)$).

We start by showing that $[\![\mathcal{A}]\!]_S(w) \leq val_S(w)$. Assume that $val_S(u) = N$. Let u' be some shortest subsequence in u ending in $\mathtt{cr}_i$ which witnesses a counter value of N for some counter i. Let w' be the corresponding subsequence in w.

In order for a letter in w' to be translated into $\mathtt{i}_i$, $\mathcal{A}$ must have been in some state $(X, \langle \gamma_j, \ldots, \gamma_k \rangle)$ with $i \geq j$ before reading a letter a in w' with $a(\gamma_i) = \mathtt{i}$ and $a(\gamma_{i'}) = \varepsilon$ for all $i' > i$. We know that u' does not contain any occurrences of $\mathtt{r}_i$ and only has $\mathtt{cr}_i$ at the last position (otherwise there would be a shorter subsequence witnessing value N). Likewise, u' does not contain any occurrences of $\mathtt{i}_{i'}$,$\mathtt{r}_{i'}$, or $\mathtt{cr}_{i'}$ for $i' > i$ (since this would induce a reset of counter i, and imply there is a shorter subsequence u'). Hence, all counters $\gamma_{i'}$ indexed

by $i' \geq i$ in the permutation are stable while $\mathcal{A}$ is reading w'. In particular, each of the N increments $\mathtt{i}_i$ in u' correspond to N increments $\mathtt{i}$ for a single counter γ_i in w'.

The fact that u' ends in $\mathtt{cr}_i$ does not imply that w' ends in $\mathtt{cr}$ for γ_i. If there is no such w' ending in $\mathtt{cr}$ for counter γ_i, then it means that no counters are checked in w so $val_S(w) = \infty$. Otherwise, at least one such subsequence must end in $\mathtt{cr}$ for γ_i, yielding value N as described above. Hence $[\![\mathcal{A}]\!]_S(w) \leq val_S(w)$.

Next, we show that $val_S(w) \leq \alpha([\![\mathcal{A}]\!]_S(w))$ for $\alpha(n) = k \cdot n^k + 1$. If $val_S(w) = \infty$, then there is no $\mathtt{cr}$ for any counter in w. But this means the run of $\mathcal{A}$ of value ∞ which never takes the check-reset transition is accepting, so $val_S(w) \leq \alpha([\![\mathcal{A}]\!]_S(w))$ as desired. Now assume that $val_S(w) = k \cdot N^k + 1$ and suppose for contradiction that $val_S(u) < N$. Consider some shortest subsequence w' of w which for some $\zeta \in \Gamma_S$ ends with $\mathtt{cr}$ for ζ and witnesses $k \cdot N^k$ increments for ζ (so there are no intermediate resets or check-resets for ζ in w').

Assume $\mathcal{A}$ is in state $(X, \langle \gamma_j, \ldots, \gamma_k \rangle)$ when starting to read w', where $\zeta = \gamma_i$ for some $i \in [j,k]$. Consider the subsequence w'' of w' (and corresponding u'' of u') on which ζ remains at this fixed index i in the permutation. On u'', there can be no resets $\mathtt{r}_{i'}$ for $i' \geq i$ or check-resets $\mathtt{cr}_{i'}$ for $i' > i$ in u', but there can be increments $\mathtt{i}_{i'}$. In fact, there can be at most $N-1$ increments $\mathtt{i}_{i'}$ before $\mathtt{i}_{i''}$ for $i'' > i'$ (otherwise it would contradict $val_S(u) < N$). Hence, these increments in u'' correspond to at most $(N-1)^k$ increments of ζ in w''.

Since there are no intermediate resets or check-resets for ζ on w', ζ can only be moved right in the permutation during the run of $\mathcal{A}$ on w'. This means the run of $\mathcal{A}$ on w' can account for $k \cdot (N-1)^k$ increments for ζ, contradicting our initial assumption. In the special case when $\zeta \in X$ when $\mathcal{A}$ starts to read w', ζ will be moved to the initial position in the permutation at the start of the run and then the reasoning proceeds as above (showing that $\mathcal{A}$ on w' can account for $k \cdot (N-1)^k + 1$ increments of ζ, again contradicting the assumption).

The history-determinism of this automaton is witnessed by the translation strategies defined by δ_n which use the check-reset transition for S-counter i (the only source of non-determinism) only when counter i would otherwise take a value higher than n.

D.2 hB and hS to hBS

We aim to prove Theorem 4. We assume that each counter type already has hierarchical actions (based on the transducers described in the previous section), and describe transducers which can make the combined B- and S-counters hierarchical in a way which preserves $\approxeq$.

We write $\mathcal{G}_{K_B}^{K_S}$ as shorthand for $\mathcal{G}(\Gamma_B, \Gamma_S)$ where $|\Gamma_B| = K_B$ and $|\Gamma_S| = K_S$, i.e. $\mathcal{G}_{K_B}^{K_S}$ is the single-state automaton over $\mathbb{C} = \mathbb{B}^{K_B} \times \mathbb{S}^{K_S}$ which outputs actions in $\mathbb{C}$ by just copying the input. Likewise, we write $\mathcal{H}_{K_B}^{K_S}$ for the hBS-automaton which will satisfy $\mathcal{H}_{K_B}^{K_S} \approxeq \mathcal{G}_{K_B}^{K_S}$

We start by describing properties and main ideas of the automaton $\mathcal{H}_{K_B}^{K_S}$. It has K_B B-counters and $(K_B+1)K_S$ S-counters. The principle of the automaton is to split the input word into sequences of S-actions from $\{\mathtt{i}, \varepsilon\}^*$ which are between resets of the B-counters. It uses one copy of the S-counter to count the number of S-increments within each sequence, and the other copy to count the number of sequences with at least one S-increment (with respect to each S-counter). If the S-value is big relative to the B-value, then the output of $\mathcal{H}_{K_B}^{K_S}$ will also have a high S-value, obtained from one of these counters. This intuition is already

used in [2], but only with boolean properties about boundedness and unboundedness; we extend the notion by paying attention to the values. Moreover, the transducer copies almost exactly the B-actions: it can only output one additional $\mathtt{r}$ between two $\mathtt{r}$'s in the input.

PROPOSITION 22. *For all $K_B, K_S \in \mathbb{N}$, the automaton $\mathcal{H}_{K_B}^{K_S}$ is history-deterministic.*

We can now use this transducer by composing it with any non-hierarchical automaton in order to transform the output and yield a hierarchical automaton.

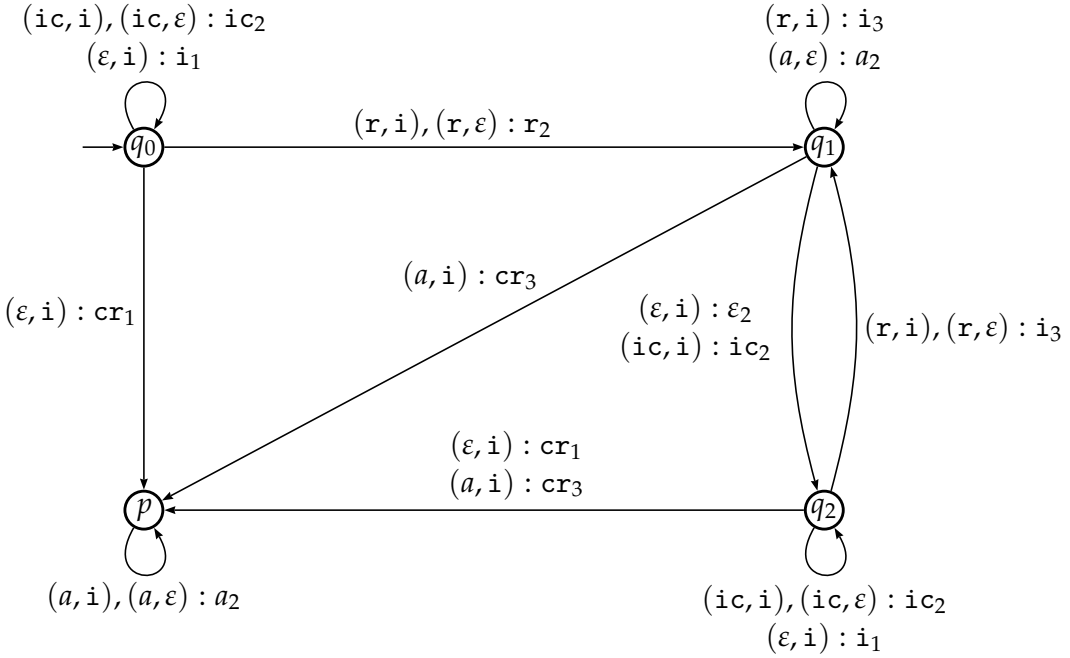
We now show how to translate a sequence of BS-actions into an equivalent hBS one, assuming that the individual B- and S-counter actions are already hierarchical.

Building $\mathcal{H}_1^1$.

We first build a transducer $\mathcal{H}_1^1$ which transforms an infinite sequence of actions in $\mathbb{B} \times \mathbb{S}$ into a hierarchical one, while guaranteeing a BS-equivalence between the input and the output.

The automaton below is $\mathcal{H}_1^1$ with the transitions labelled by letters from $\mathbb{B} \times \{\mathtt{i}, \varepsilon\}$ (other transitions are described later, in order to lighten the picture). The automaton has 3 hierarchical counters S_3, B_2, S_1 and hierarchical actions are labeled by counter number, for example action $\mathtt{ic}_2$ stands for $(\mathtt{r}, \mathtt{ic}, \varepsilon)$ which does not change S_3, increments and checks B_2, and resets S_1. We write a for any action of $\mathbb{B}$. For all states s, there is a loop $s \xrightarrow{(\varepsilon,\varepsilon):\varepsilon_1} s$.

$\mathcal{H}_1^1$ can be considered as an automaton on either finite or infinite words over $\mathbb{C}$, using the fact that if $w \in \mathbb{C}^\omega$ is accepting for both Büchi conditions, then we have $[\![\mathcal{H}_1^1]\!]_B(w) = \sup\{[\![\mathcal{H}_1^1]\!]_B(u) : u \text{ prefix of } w\}$, and for all $m \in \mathbb{N}$, we also have that $[\![\mathcal{H}_1^1]\!]_S^B(m)(w) = \inf\{[\![\mathcal{H}_1^1]\!]_S^B(m)(u) : u \text{ prefix of } w\}$. Therefore correctness of $\mathcal{H}_1^1$ over finite words implies correctness over infinite words.



We add the following transitions:

(1) $q_0 \xrightarrow{(a,\mathtt{r}):a_2} q_0$,
(2) for $x \in \{1,2\}$, $q_x \xrightarrow{(a,\mathtt{r}):\mathtt{r}_3} q_0$,
(3) $p \xrightarrow{(a,\mathtt{r}),(a,\mathtt{cr}):a_2} q_0$.

Remark that the automaton is built in such a way that p much be reached when reading a `cr` on the input, but in order to reach p, we have to perform a `cr` on one of the S-counters. The principle of the automaton is that on a sequence of `i` and ε on the input S-counter, γ_3 counts the number of intervals between two `r`'s for B which contain at least one `i`, whereas γ_1 counts the number of `i`'s during an ε-sequence for B. The bounded number of `ic`'s will ensure the semantic is correct.

THEOREM 23. *$\mathcal{H}_1^1$ is BS-equivalent to $\mathcal{G}_1^1$ which just copies the input actions.*

PROOF. First, let us show that $[\![\mathcal{H}_1^1]\!]_B \approx [\![\mathcal{G}_1^1]\!]_B$. The only transitions of $\mathcal{H}_1^1$ which do not reproduce the action of the B-counter are the ones defined in (2), and the transitions from q_1 or q_2 to p labelled $(a,\mathtt{i}) : \mathtt{cr}_3$ (all of these transitions perform a reset on counter B_2 instead of action a).

However, to reach these transitions from q_0, one must have seen a reset of the B-counter before (transitions from q_0 to q_1), and moreover we go back to state q_0 by taking only one such transition. We can insert (by replacing a letter) at most one reset between two resets of the initial sequence, so $[\![\mathcal{H}_1^1]\!]_B \approx_\alpha [\![\mathcal{G}_1^1]\!]_B$, with $\alpha(m) = 2m+1$.

We will now show the remaining part of $[\![\mathcal{H}_1^1]\!]_{BS} \approx_\alpha [\![\mathcal{G}_1^1]\!]_{BS}$.

Let $m \in \mathbb{N}$ be fixed, we want to show that there exists β_m such that

- $[\![\mathcal{H}_1^1]\!]_S^B(m) \preccurlyeq_{\beta_m} [\![\mathcal{G}_1^1]\!]_S^B(\alpha(m))$ (1)
- $[\![\mathcal{G}_1^1]\!]_S^B(m) \preccurlyeq_{\beta_m} [\![\mathcal{H}_1^1]\!]_S^B(\alpha(m))$ (2)

Part (1) is quite straightforward: notice that we do increments on counter S_1 or S_3 of $\mathcal{H}_1^1$ only when there is an increment for an S-counter in u. Moreover, every `cr` in u can be matched to an earlier `cr` on counter S_1 or S_3 in $\mathcal{H}_1^1$. Therefore, $[\![\mathcal{H}_1^1]\!]_S^B(m) \leq [\![\mathcal{G}_1^1]\!]_S^B(m) \leq [\![\mathcal{G}_1^1]\!]_S^B(\alpha(m))$.

For part (2), we proceed by contradiction, and assume that there is a set U of finite words on alphabet $\mathbb{C}$ such that $\{[\![\mathcal{H}_1^1]\!]_S^B(\alpha(m))(u) : u \in U\}$ is bounded by some $N \in \mathbb{N}$, but $\{[\![\mathcal{G}_1^1]\!]_S^B(m)(u) : u \in U\}$ is unbounded.

Let $u \in U$ such that $[\![\mathcal{G}_1^1]\!]_S^B(m)(u) > ((N+1)m+1)N$. In particular, $[\![\mathcal{G}_1^1]\!]_S^B(m)(u)$ is not 0 so it is equal to $val_S(u)$, and $val_B(u) \leq m$.

We write $u = (u_B, u_S)$, and factorize u_S in $u_1 b_1 u_2 b_2 \dots u_k b_k u_{k+1}$ such that for any j, $b_j \in \{\mathtt{r}, \mathtt{cr}\}$ and $u_j \in \{\mathtt{i}, \varepsilon\}^*$. We do the corresponding factorization $u_B = v_1 a_1 v_2 a_2 \dots v_k a_k v_{k+1}$ by matching the positions of a_j and b_j.

Let us remark that a run ρ on u can be written $\rho_1 \dots \rho_k \rho'$ such that for all j, ρ_j is the subrun of ρ over $(v_j a_j, u_j b_j)$. By definition of $\mathcal{H}_1^1$ (namely, the transitions described in (1)-(3) above), ρ_j starts and ends in state q_0 for all j.

Moreover, let $n = [\![\mathcal{H}_1^1]\!]_S^B(\alpha(m))(u) \leq N$, there is an optimal run ρ of $\mathcal{H}_1^1$ on u with $val_S(\rho) = n$. "Optimal" means here that for all j, $val_S(\rho_j)$ is maximal: there is no run ρ_j' over $(v_j a_j, u_j b_j)$ starting in q_0 with $val_S(\rho_j') > val_S(\rho_j)$. We can take ρ optimal because the subruns ρ_j are independent from each other.

Let J be the set of indices j such that $b_j = \mathtt{cr}$. For any $j \in J$, we have $val_S(u_j\mathtt{cr}) \geq val_S(u) = [\![\mathcal{G}_1^1]\!]_S^B(m)(u) > ((N+1)m+1)N$. Let $j \in J$ such that $val_S(\rho_j) = n$, which exists because $val_S(\rho) = n$. We have $q_0 \xrightarrow{(v_j,u_j)}{}^* p \xrightarrow{(b_j,\mathtt{cr})} q_0$.

Let $v_j = w_0\mathtt{r}w_1\mathtt{r}\dots\mathtt{r}w_s$ with $w_l \in \{\mathtt{ic},\varepsilon\}^*$ for all l, and let $u_j = x_0d_0x_1\dots d_{s-1}x_s$ be the corresponding factorization of u_j, i.e. for all l, $|x_l| = |w_l|$ and $|d_l| = 1$.

If $s = 0$, then the transition from q_0 to q_1 is not taken during ρ_j. But when the transition towards p is taken, we perform action $\mathtt{cr}_1$. This means the counter S_1 must have value n, after n consecutive readings of $(\varepsilon,\mathtt{i})$. Assume that there are at most n consecutive $(\varepsilon,\mathtt{i})$ in (v_j,u_j). We know that $|v_j|_{\mathtt{ic}} \leq m$ so we get $|u_j|_{\mathtt{i}} \leq (n+1)m$. But $val_S(u_j\mathtt{cr}) > ((N+1)m+1)(\alpha(m)+1) > (n+1)m$ so it is absurd. It means that there is strictly more than n consecutive $(\varepsilon,\mathtt{i})$ in (v_j,u_j), which contradicts the optimality of ρ, since we can build a run ρ_j' on (v_ja_j,u_jb_j) with $val_S(\rho_j') > n = val_S(\rho_j)$.

It remains to treat the case $s > 1$. As before, we must have $\sup_i(|x_i|_{\mathtt{i}}) \leq (n+1)m$, otherwise ρ would not be optimal. Consequently, for all i, $|x_id_i|_{\mathtt{i}}) \leq (n+1)m+1$. Notice that the value of the counter S_3 while reading (w_i,x_i) is equal to the size of $J_i = \{l \leq i, |x_ld_l|_{\mathtt{i}} > 0\}$. Therefore we must have $|J_s| \leq n$ by optimality of ρ, otherwise we could build a run of value greater than ρ_j by taking a transition labeled $\mathtt{cr}_3$ towards p at the end of (w_j,x_j).

Finally $|u_j|_{\mathtt{i}} \leq (|x_jd_j|_{\mathtt{i}})|J_s| \leq ((n+1)m+1)n \leq ((N+1)m+1)N < [\![\mathcal{G}_1^1]\!]_S^B(m)(u)$. It is true for all j, which is in contradiction with the definition of $[\![\mathcal{G}_1^1]\!]_S^B$.

We can conclude there is no such set U, so $[\![\mathcal{G}_1^1]\!]_S^B(m) \preccurlyeq_{\beta_m} [\![\mathcal{H}_1^1]\!]_S^B(\alpha(m))$, for $\beta_m(n) = ((n+1)m+1)n$.

THEOREM 24. *$\mathcal{H}_1^1$ is history-deterministic.*

PROOF. Let us first notice that the only non-determinism in $\mathcal{H}_1^1$ occurs while reading $\mathtt{i}$ on the $\mathbb{S}$ component (transitions going to p). We define δ_n to be the strategy taking transitions going to p if we have reached n on some S-counter and then see another $\mathtt{i}$ action (this condition depends only on the past word u). This already defines a unique run ρ_u of $\mathcal{H}_1^1$ on every word u such that $val_S(\rho_u) \geq n$.

Moreover, if there is a run ρ of $\mathcal{H}_1^1$ over u with $val_S(\rho) \geq n$, then the run driven by δ_n exists (and, in fact, is the optimal run), which concludes the proof that $\mathcal{H}_1^1$ is history-deterministic.

Extension of $\mathcal{H}_1^1$ for several hierarchical counters.

Let $K_B, K_S > 0$, we define $\mathbb{C}_{K_B}^{K_S} \subset (\mathbb{B})^{K_B} \times (\mathbb{S})^{K_S}$ to be the set of actions that can be made by a hierarchical B-automaton together with a hierarchical S-automaton. $\mathbb{C}_{K_B}^{K_S}$ is composed of letters of the form $((\mathtt{r},\dots,\mathtt{r},a,\varepsilon,\dots,\varepsilon),(\mathtt{r},\dots,\mathtt{r},b,\varepsilon,\dots,\varepsilon))$ with $(a,b) \in \mathbb{B} \times \mathbb{S}$.

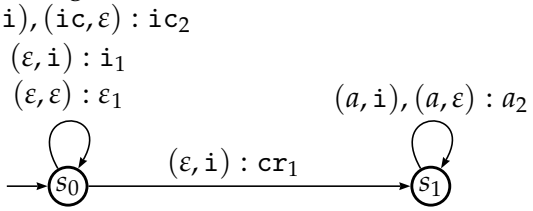
For any $K_B, K_S > 0$, we now want to build $\mathcal{H}_{K_B}^{K_S}$, the generalization of $\mathcal{H}_1^1$ to alphabet $\mathbb{C}_{K_B}^{K_S}$, such that it is BS-equivalent to the automaton $\mathcal{G}_{K_B}^{K_S}$ on the same alphabet, with one state, which just copies the input actions to its counters. As before, we want $\mathcal{H}_{K_B}^{K_S}$ to be hierarchical and history-deterministic.

We first define by induction the automaton $\mathcal{H}_{K_B}^1$ for all $K_B > 0$.

Building $\mathcal{H}^1_{K_B}$ for all $K_B > 0$.

First notice that $\mathcal{H}^1_1 = \langle \mathbb{C}^1_1, Q^1, \{q^1_0\}, \Gamma^1_B, \Gamma^1_S, \Delta^1 \rangle$ is already built in a correct way. Now assume $\mathcal{H}^1_{K_B} = \langle \mathbb{C}^1_{K_B}, Q^{K_B}, \left\{q^{K_B}_0\right\}, \Gamma^{K_B}_B, \Gamma^{K_B}_S, \Delta^{K_B} \rangle$ is built and correct, and we want to build $\mathcal{H}^1_{K_B+1}$.

We need $\mathcal{C} = \langle \{\mathtt{ic}, \varepsilon\} \times \{\mathtt{i}, \varepsilon\}, Q_{\mathcal{C}}, \{s_0\}, \{\gamma_B\}, \{\gamma_S\}, \Delta_{\mathcal{C}} \rangle$, an auxiliary BS-automaton defined by this diagram:



which has two new hierarchical counters, $\gamma_B > \gamma_S$.

We define $\mathcal{H}^1_{K_B+1} = \langle \mathbb{C}^1_{K_B+1}, Q^{K_B+1}, \left\{q^{K_B+1}_0\right\}, \Gamma^{K_B+1}_B, \Gamma^{K_B+1}_S, \Delta^{K_B+1} \rangle$. Let $Q^{K_B+1} = Q^{K_B} \times Q_{\mathcal{C}}$, $q^{K_B+1}_0 = (q^{K_B}_0, s_0)$. We take $\Gamma^{K_B+1}_B = \Gamma^{K_B}_B \cup \{\gamma_B\}$ and $\Gamma^{K_B+1}_S = \Gamma^{K_B}_S \cup \{\gamma_S\}$ such that $\gamma > \gamma_B > \gamma_S$ for all $\gamma \in \Gamma^{K_B}_B \cup \Gamma^{K_B}_S$. The hierarchy of counters is inherited from $\mathcal{H}^1_{K_B}$ and $\mathcal{C}$, and any action on counters of $\mathcal{H}^1_{K_B}$ performs resets on counters of $\mathcal{C}$.

Finally, Δ^{K_B+1} is defined as follows.

(C) for all $s \xrightarrow{(a,b):\sigma} s'$ in $\Delta_{\mathcal{C}}$ with $a \in \mathbb{B}$ and $b \in \mathbb{S}$, and all $q \in Q^{K_B}$, we add transition $(q,s) \xrightarrow{(\varepsilon_{K_B}a,b):\sigma} (q,s')$ in Δ^{K_B+1}.

(Hx) for all $q \xrightarrow{(a,b):\sigma} q'$ in Δ^{K_B} with $a \in \mathbb{B}^x$ and $b \in \mathbb{S}^1$, and all $s \in Q_{\mathcal{C}}$, we add transition $(q,s) \xrightarrow{(a\mathtt{r},b):\sigma} (q',s_0)$ in Δ^{K_B+1}.

(s1) for all $q \in Q^{K_B}$, and $a \in \mathbb{B}^{K_B+1}$, we add transition $(q,s_1) \xrightarrow{(a,\mathtt{cr}):a} (q^{K_B}_0, s_0)$

where $\varepsilon_{K_B}a$ stands for $(\varepsilon, \ldots, \varepsilon, a)$, $a\mathtt{r}$ stands for $(a,\mathtt{r})$ with a being an action on K_B hierarchical B-counters, and action σ is performed on the same counter as before, i.e. γ_B or γ_S in the case (C), and one of the counters of $\Gamma^{K_B}_B$ or $\Gamma^{K_S}_S$ in case (Hx).

Let us show that $\mathcal{H}^1_{K_B+1}$ is BS-equivalent to $\mathcal{G}^1_{K_B+1}$.

We first have to show $[\![\mathcal{H}^1_{K_B+1}]\!]_B \approx_\alpha [\![\mathcal{G}^1_{K_B+1}]\!]_B$, we will again use $\alpha(m) = 2m+1$. By induction, we can show that as in the preceding proof, for each counter, at most one $\mathtt{r}$ is added for each $\mathtt{r}$ of the input sequence, so $[\![\mathcal{H}^1_{K_B+1}]\!]_B \leq val_B$ which implies the result.

We now need to show that that for all m, there exists β_m such that :

- $[\![\mathcal{H}^1_{K_B+1}]\!]^B_S(m) \preccurlyeq_{\beta_m} [\![\mathcal{G}^1_{K_B+1}]\!]^B_S(\alpha(m))$ (1)
- $[\![\mathcal{G}^1_{K_B+1}]\!]^B_S(m) \preccurlyeq_{\beta_m} [\![\mathcal{H}^1_{K_B+1}]\!]^B_S(\alpha(m))$ (2)

Part (1) is still straightforward: increments on B counters always corresponds to increments corresponding counters of the input words, and every input $\mathtt{cr}$ leads to an output $\mathtt{cr}$ after less increments.

It remains to show part (2). Let $m \in \mathbb{N}$. Let $u = (u_B, u_S) \in (\mathbb{B}^{K_B+1} \times \{\mathtt{i}, \varepsilon\})^*(\mathbb{B}^{K_B+1} \times \{\mathtt{cr}\})$ such that $[\![\mathcal{G}^1_{K_B+1}]\!]^B_S(m)(u) > N$ and $[\![\mathcal{H}^1_{K_B+1}]\!]^B_S(\alpha(m))(u) \leq n$. We restrict u to this language, because as in the preceding case where $K_B = 1$, the automaton is reset to its initial state when it reads a $\mathtt{cr}$, and the runs over these factors are independent of each

other. Resetting the S-counter with $\mathtt{r}$ corresponds to ignoring u (the automaton can avoid transitions doing $\mathtt{cr}$ on a S-counter during u), and also resets the automaton. The general case follows directly from this one.

We write $u_B = u_1 v_1 u_2 \dots u_k v_k a$ where for all $i \leq k$, $u_i \in (\mathbb{B}^{K_B} \times \{\mathtt{r}\})^*$ and $v_i \in (\{\varepsilon\}^{K_B} \times \{\mathtt{ic}, \varepsilon\})^*$, only u_1 and v_k are allowed to be ε for unicity of this factorisation, and a is just the last letter of u_B. We do the corresponding factorization $u_S = u'_1 v'_1 u'_2 \dots u'_k v'_k \mathtt{cr}$.

Let ρ be a run of $\mathcal{H}^1_{K_B+1}$ on u, We can write $\rho = \rho_1 \tau_1 \rho_2 \dots \tau_{k-1} \rho_k$ where $\rho_{K_B} = \rho_1 \dots \rho_k$ is a run of $\mathcal{H}^1_{K_B}$ over $u_1 \dots u_k$, and for all $i < k$, τ_i is a run of $\mathcal{C}$ over v_i.

By induction hypothesis, there exists $\beta^m_{K_B}$ such that $|u_1 \dots u_k|_{\mathtt{i}} \leq_{\beta^m_{K_B}} val_S(\rho_{K_B})$. It is straightforward to show that for $\beta^m_{\mathcal{C}}(n) = (m+1)n$, we have for all $i < k$, $|u_i|_{\mathtt{i}} \leq_{\beta^m_{\mathcal{C}}} val_S(\tau_i)$. ($val_S$ meaning here the maximum value reached by a S-counter, not necessarily checked).

We get $val_S(u) \leq_{\beta^m_{K_B+1}} val_S(\rho)$, for $\beta^m_{K_B+1} = \max(\beta^m_{K_B}, \beta^m_{\mathcal{C}})$, which shows the wanted result.

By induction, for all $K_B > 0$, we have built $\mathcal{H}^1_{K_B}$, hierarchical and BS-equivalent to $\mathcal{G}^1_{K_B}$. The history-determinism is straightforward, the strategy for value n being the same as before, i.e. taking transition performing $\mathtt{cr}$ when we want to increment a S-counter already at value n.

Remark that by induction, for all $K_B > 0$, the automaton $\mathcal{H}^1_{K_B}$ has 2^{K_B+1} states and $2K_B + 1$ counters.

Building $\mathcal{H}^{K_S}_{K_B}$ for any $K_B > 0$ and $K_S > 0$.

We fix the values of K_B and K_S, and describe informally the automaton $\mathcal{H}^{K_S}_{K_B}$.

Let $\mathcal{H}^1_{K_B}$ be as previously defined. It has counters $S_{K_B+1} B_{K_B} S_{K_B} \dots B_1 S_1$.

$\mathcal{H}^{K_S}_{K_B}$ will have the same B-counters as $\mathcal{H}^1_{K_B}$ i.e. $(\gamma_i)_{i \in [1,K_B]}$, and S-counters $(\gamma^j_i)_{i \in [1,K_B+1], j \in [1,K_S]}$.

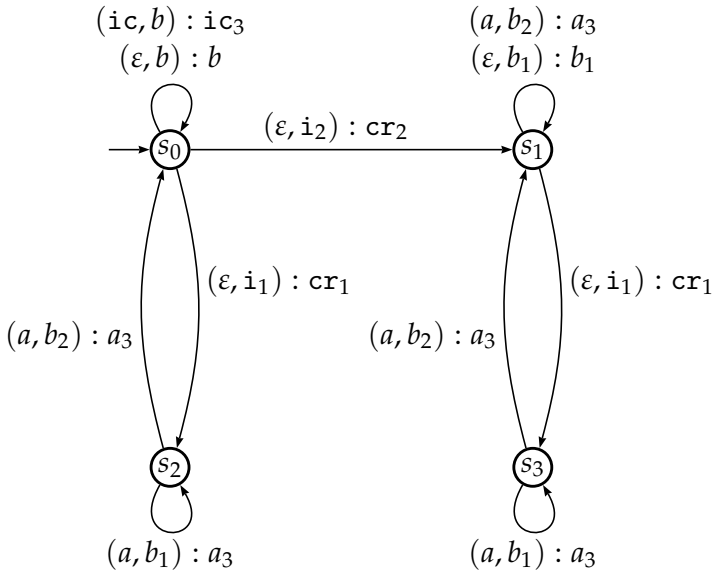
We define the hierarchical order $\gamma^j_i < \gamma^{j'}_{i'}$ by lexicographic order on (i,j), and for all i, j, $\gamma_i > \gamma^j_i$. Finally, for all j, $\gamma^j_{K_B+1} > \gamma_i$.

Let γ_i be a B-counter of $\mathcal{H}^{K_S}_{K_B}$. The principle of $\mathcal{H}^1_{K_B}$ is to count the number of increments of the S-counter below during ε-sequences of γ_i, and to perform $\mathtt{cr}$ if it becomes too high (thereby going to a state where a later $\mathtt{cr}$ on the input is allowed). The highest S-counter counts the number of $\{\mathtt{ic}, \varepsilon\}$-sequence of γ_{K_B}-actions that start with $\mathtt{r}$ and contain $\mathtt{i}$ for the S-counter.

We have duplicated each S-counter γ'_i into K_S ones γ^j_i which will play the some role, but relative to their original counter γ_j.

The K_S S-counters being originally hierarchical, we will satisfy the global hierarchical condition for $\mathcal{H}^{K_S}_{K_B}$.

For instance here is the automaton $\mathcal{C}_2$ which plays the role of $\mathcal{C}$ in the case $K_S = 2$. Automaton $\mathcal{C}_2$ is defined over alphabet $\{\mathtt{ic}, \varepsilon\} \times \{\mathtt{i}_1, \mathtt{i}_2, \varepsilon_2\}$, and has 2 S-counters and one B-counter $\gamma_1 < \gamma_2 < \gamma_B$.



The automaton is allowed to see a cr for γ_1 in s_2 and s_3, and a cr for γ_2 in s_1 and s_3. As before we can show that $\mathcal{H}^{K_S}_{K_B}$ is correct and history-deterministic.

$\mathcal{H}^{K_S}_{K_B}$ has K_B B-counters, $(K_B + 1)K_S$ S-counters.

E Construction of Quasi-Weak B-Automaton $\mathcal{B}$

E.1 Proof of Proposition 9

Let $\mathcal{U}$ (respectively, $\mathcal{U}'$) be non-deterministic B-Büchi (respectively, S-Büchi). Let $\mathcal{A} = \langle Q_{\mathcal{A}}, \mathbb{A}, q_0, F_B, F_S, \Gamma_B, \Gamma_S, \delta \rangle$ be a non-deterministic hBS-automaton such that $\mathcal{A} \cong \mathcal{U} \times \mathcal{U}'$. We assume there is a cost trap for $\mathcal{A}$, and we want to show that $[\![\mathcal{U}']\!] \not\leq [\![\mathcal{U}]\!]$.

Let $m := (|Q_{\mathcal{A}}| + 2)^{|\Gamma_S|+1}$. Let E_m be a frontier, $\{e_i^\pi : 0 \leq i \leq m, \pi \in \{0,1\}^\omega\}$ be a set of nodes, t be an input tree, and R be a run of $\mathcal{A}$ on t witnessing the cost trap.

By definition of a cost trap, we have that $val_S(R) > |Q_{\mathcal{A}}|$, and for all branches π, $e_0^\pi < \cdots < e_m^\pi \in E_m$ are nodes of π such that for all $0 \leq i < m$ the portion of R corresponding to $[e_i^\pi, e_{i+1}^\pi)$ is a block, i.e. contains states from both F_B and F_S, and for all $\gamma \in \Gamma_B$, if γ is incremented in this portion then it is also reset in this portion.

Moreover, if branches π_1 and π_2 share some prefix up to position y and $x < y$ is the first position with $e_i^{\pi_1} = x$ and $e_i^{\pi_2} \neq x$ then $e_i^{\pi_2} > y$ (we will call this the *nesting condition*). In this case we will say that $\pi_1 <_{nest} \pi_2$ and y is a *nesting node*. If π_1 and π_2 do not disagree on the e_i's on their common prefix, we will say that $\pi_1 \approx_{nest} \pi_2$. The idea is that if $\pi_1 <_{nest} \pi_2$, then pumping the blocks from π_2 will not damage any of the blocks from π_1.

The first step will be to build a single tree t' such that there is some n with $[\![\mathcal{A}]\!]_B(t') \leq n < \infty$, and also $[\![\mathcal{A}]\!]^B_S(n)(t') > |Q_{\mathcal{A}}|$ (we remind that $[\![\mathcal{A}]\!]^B_S$ is a function $\mathbb{N} \to \mathcal{T}_{\mathbb{A}} \to \mathbb{N}_\infty$).

Let t_{E_m} be the finite tree obtained from t by removing all nodes after the frontier E_m.

The tree t' will be obtained from t_{E_m} by an infinite pumping of some blocks on every branch.

Let A be the set of all branches of t_{E_m}. A is a finite set, and is partially ordered by $<_{nest}$. We inductively build sets $(A_j)_{j \in \mathbb{N}}$ in the following way : for all j, A_j is the set of branches

of t_{E_m} that are maximal for $<_{nest}$ in $A \setminus (\bigcup_{j'<j} A_{j'})$. Let $p \in \mathbb{N}$ be such that $A_p \neq \emptyset$ and for all $j > p$, $A_j = \emptyset$. Then $A_1, \ldots, A_p$ form a partition of A, and for all $j \in [1,p]$, for all $\pi_1, \pi_2 \in A_j$, we have $\pi_1 \approx_{nest} \pi_2$. Notice that for all $j \in [1,p]$ and all $i \in [1,m]$, the set $\{e_i^\pi : \pi \in A_j\}$ is a partial frontier of t_{E_m} (more precisely, it is the intersection of a frontier of t_{E_m} with A_j).

We can now adapt the technique of Rabin in [12], but applying it only on the branches of A_1 (for now). Let $N := |Q_\mathcal{A}|$. In the following, S-values will be approximated by giving value $N+1$ if the value is larger than N. Unlike [12], when we pump we must be careful that we do not reduce the S-value below N.

We define sets $H_m, \ldots, H_0$ by induction : we set $H_m := Q_\mathcal{A} \times [0, N+1]^{\Gamma_S}$, and $(q,\eta) \in H_{i-1}$ if $(q,\eta) \in H_i$ and there is a finite tree t_f contained in t_{E_m} and a partial run R_f of $\mathcal{A}$ over t_f starting from state q and valuation η for the S-counters, such that

- every branch of R_f is a block,
- on every leaf of t_f, the state and S-counters values configuration (q', η') of $\mathcal{A}$ belongs to H_i.

Notice that some nodes of t_f are allowed to have only arity 1 (those will correspond to nesting nodes of t_{E_m}). The idea is that we want to be able to reach a H_i-frontier by block branches, when starting from a node in H_{i-1} (which is also in H_i).

We have $H_0 \subseteq H_1 \subseteq \cdots \subseteq H_m$, and moreover, the run R on branches A_1 witnesses the fact that for all branch $\pi \in A_1$, for all $i \in [0,m]$, the configuration (q,η) of $\mathcal{A}$ in the run R at node e_i^π always belongs to H_i. Moreover, there is a finite tree t_0 such that $\mathcal{A}$ has a run over t_0 starting in $(q_0, 0)$ and ending in a configuration belonging to H_0 on every leaf (t_0 is a prefix of t_{E_m}, and this run is a prefix of the run R). Remark that the approximation of S-values up to N is not a problem, since counters can only increment or reset, so coherence is kept on values less than N on one hand (by remembering exact values), and values more than N on the other hand. Moreover, no block from R can check a S-value less than N, because $val_S(R) > N$.

But $|H_m| = N\dot{(}N+2)^{|\Gamma_S|} \leq m$, so there must be $i \in [0, m-1]$ such that $H_i = H_{i+1}$. It means that from any configuration of H_i, we can find a finite tree t_f contained in t_{E_m}, and a partial run of $\mathcal{A}$ over t_f (contained in R), such that any branch of this run is a block, and the configuration on any leaf is again in H_i. This will allow a pumping, creating an infinite tree and a run with infinitely many H_i-frontiers. We want the B-value of this run to stay low, and that is why the definition of blocks include a constraint on B-actions : for all B-counter γ, if a block increments γ, then it also resets γ. In fact, when appending two blocks, the worst case that can happen is when the first block starts with a reset and then make some increments, and the second one make some increments and then resets.

This is why we define $n_{\text{blocks}} := \max\{val_B(uv) : u, v \text{ are blocks in } R \text{ from } H_i \text{ to } H_i\}$, to be the maximal B-value obtained by appending two blocks. Notice than appending more than two blocks cannot increase this value. We also define n_{pref} to be the B-value of the partial run reaching the first H_i-frontier from the root in the run R. Finally, let $n_1 := n_{\text{blocks}} + n_{\text{pref}}$.

This allows us to build an infinite (but not complete) tree t_1, and a partial run R_1 of $\mathcal{A}$ over t_1, by using R to reach the H_i-frontier (nodes e_i^π for $\pi \in A_1$), and then repeating infinitely many finite trees witnessing partial runs from H_i-nodes to H_i-frontiers. Note that

every infinite play π of R_1 is B-accepting with $val_B(\pi) \leq n_1$, and S-accepting with $val_S(\pi) > N$, by the block condition and the fact that no S-value less than N is checked during R.

The tree t_1 also contains infinitely many finite branches, duplicated from $A_2, \dots, A_p$. Let us call A'_2 the infinite set of finite branches coming from A_2. The nesting condition implies that pumping branches of A_1 did not split blocks of the other A_i's, so we can now define the sets $H_0, \dots H_m$ as before, and pump all the branches of A'_2 simultaneously, building a tree t_2 satisfying the same properties as t_1, but with more infinite branches.

Iterating p times this process yields a complete binary infinite tree $t' := t_p$, and some $n := n_1 + n_2 + \cdots + n_p$ (only depending on the finite set of blocks from the run R on t_{E_m}) such that there is a run R' of $\mathcal{A}$ on t' witnessing both $[\![\mathcal{A}]\!]_B(t') \leq n$, and $[\![\mathcal{A}]\!]^B_S(n)(t') > N$ (i.e. the run R' is B-accepting with B-value less than n, and S-accepting with S-value strictly above N).

We will now build from t' an infinite sequence of infinite trees $(t'_d)_{d\in\mathbb{N}}$, such that for all $d \in \mathbb{N}$, $[\![\mathcal{A}]\!]_B(t'_d) \leq n$, and $[\![\mathcal{A}]\!]^B_S(n)(t'_d) > N + d$.

In order to do that, consider the run R' on t', and more specifically the actions on the S-counters. In order for R' to have S-value more than N, any action $\mathtt{cr}_\gamma$ for $\gamma \in \Gamma_S$ must occur when the value of γ is at least $N + 1$, so the $\mathtt{cr}$ must be immediately preceded by a sequence of at least $N + 1$ increments of γ : this sequence may contain ε but cannot contain any $\mathtt{r}$ or $\mathtt{cr}$ (again for counter γ).

During such a sequence and by choice of N, there must be a path u_γ from some $q \in Q_\mathcal{A}$ to the same state q, starting with one increment and containing no reset ($\mathtt{r}$ or $\mathtt{cr}$) of γ.

Recall that we are using hierarchical BS-actions. Let us look at the behaviours of other counters during such a path u_γ.

- If γ' is a B- or S-counter higher than γ in the hierarchy, touching it resets γ, so γ' is not touched at all during u_γ : this path contains only action ε for $\gamma' > \gamma$.
- If γ' is a B- or S-counter lower than γ, then u_γ starts with action $\mathtt{r}$ for γ'.

In any case, repeating u_γ several times instead of one does not affect the values of other B- and S-counters,

Moreover, two such paths u_γ and $v_{\gamma'}$ cannot intersect if $\gamma \neq \gamma'$. If on the contrary $\gamma = \gamma'$, and the two paths share a prefix, then repeating one can inflate the other, but never changes the properties we are interested in : it remains a path from some $q \in Q_\mathcal{A}$ to q, starting with an action $\mathtt{i}_\gamma$ and without any reset for γ. We will call t'_d the infinite tree obtained from t' where for all S-counter γ, for all position x where action $\mathtt{cr}_\gamma$ is done in R', the path u_γ relative to this position x is repeated d times. The run R' induces a run R'_d for each t'_d, which is still B-accepting and S-accepting (accepting states still appear infinitely often), has B-value less than n, but where at least d increments have been added before each action $\mathtt{cr}$ of any counter, so $val_S(R_d) > N + d$.

We are now ready to observe the contradiction. Let α and $(\beta_i)_{i\in\mathbb{N}}$ be corrections functions witnessing $\mathcal{A} \cong \mathcal{U} \times \mathcal{U}'$. In particular $[\![\mathcal{A}]\!]_B \approx_\alpha [\![\mathcal{U} \times \mathcal{U}']\!]_B = [\![\mathcal{U}]\!]_B$ ($\mathcal{U}'$ does not play any role in the B-semantic of $\mathcal{U} \times \mathcal{U}'$). It implies that for all $d \in \mathbb{N}$, $[\![\mathcal{U}]\!]_B(t'_d) \leq \alpha(n)$.

But we also have $[\![\mathcal{A}]\!]^B_S(n) \preccurlyeq_{\beta_n} [\![\mathcal{U} \times \mathcal{U}']\!]^B_S(\alpha(n))$. Let $d \in \mathbb{N}$, we have in particular, $N + d \leq \beta_n([\![\mathcal{U} \times \mathcal{U}']\!]^B_S(\alpha(n))(t'_d))$.

But $[\![\mathcal{U}\times\mathcal{U}']\!]_S^B(\alpha(n))(t'_d) = \sup\{val_S(R'') : R''$ is an accepting run of $\mathcal{U}\times\mathcal{U}'$ on t'_d with $val_B(R'') \leq \alpha(n)\}$. Since we know that $[\![\mathcal{U}\times\mathcal{U}']\!]_B(t'_d) \leq \alpha(n)$, there is an S-accepting run R'' of $\mathcal{U}\times\mathcal{U}'$ on t'_d witnessing $\beta_n([\![\mathcal{U}\times\mathcal{U}']\!]_S^B(t'_d)) \geq N+d$. This means $\beta_n(val_S(R'')) \geq N+d$. But R'' induces an S-accepting run $R_{\mathcal{U}'}$ of $\mathcal{U}'$ over t'_d with $\beta_n(val_S(R_{\mathcal{U}'})) \geq N+d$.

We get that for all $d\in\mathbb{N}$, $\beta_n([\![\mathcal{U}']\!]_S(t'_d)) \geq N+d$, so $[\![\mathcal{U}']\!]_S$ is unbounded over the set $\{t'_d : d\in\mathbb{N}\}$ while $[\![\mathcal{U}]\!]_B$ is bounded over the same set.

This shows that a cost trap implies a contradiction with $[\![\mathcal{U}']\!]_S \preccurlyeq [\![\mathcal{U}]\!]_B$.

E.2 Construction of $\mathcal{B}$

Given both a non-deterministic B-Büchi automaton $\mathcal{U} = \langle Q_{\mathcal{U}}, \mathbb{A}, q_0^{\mathcal{U}}, F_B^{\mathcal{U}}, \Gamma_{\mathcal{U}}, \Delta_{\mathcal{U}}\rangle$ and a non-deterministic S-Büchi automaton $\mathcal{U}' = \langle Q_{\mathcal{U}'}, \mathbb{A}, q_0^{\mathcal{U}'}, F_S^{\mathcal{U}'}, \Gamma_{\mathcal{U}'}, \Delta_{\mathcal{U}'}\rangle$ such that $[\![\mathcal{U}]\!]_B \approx [\![\mathcal{U}']\!]_S$, we want to build a quasi-weak B-automaton $\mathcal{B}$ recognizing the same cost function as $\mathcal{U}$ and $\mathcal{U}'$.

Let $\mathcal{H} = \langle Q_{\mathcal{H}}, \mathbb{C}, q_0^{\mathcal{H}}, \Gamma_B, \Gamma_S, F_B, F_S, \Delta_{\mathcal{H}}\rangle$ be the transducer $\mathcal{H}(\Gamma_{\mathcal{U}}, \Gamma_{\mathcal{U}'})$ from Theorem 4.

We build the desired quasi-weak B-automaton $\mathcal{B}$ with the following set of states :

$$Q := Q_{\mathcal{U}}\times Q_{\mathcal{U}'}\times Q_{\mathcal{H}}\times (([1,m]\times\{\epsilon,\bot,\top\}\times\{\varepsilon,\mathtt{ic},\mathtt{r}\})^{[0,|\Gamma_B|]}\cup\{q_\top\})$$

with initial state $(q_0^{\mathcal{U}}, q_0^{\mathcal{U}'}, q_0^{\mathcal{H}}, \{(1,\epsilon,\varepsilon)\}^{[0,|\Gamma_B^{\mathcal{H}}|]})$.

In fact, the only useful states will be those of Q_{use} : an element $(q_1,q_2,q_3,(i,j,z))$ is in Q_{use} if for all $k,k'\in[0,|\Gamma_B^{\mathcal{H}}|]$, $k<k'$ implies $i(k)\leq i(k')$ and $j(k)\leq j(k')$, for the order $\epsilon<\bot<\top$. Moreover any $(q_1,q_2,q_3,q_\top)$ is in Q_{use}.

Consider $\mathcal{B}$ acting on a tree t. The idea is that Eve selects a run of $\mathcal{U}$ on t and Adam selects a run of $\mathcal{U}'$ on t (technically this is done one move at a time). The B-actions output are exactly those from the run of $\mathcal{U}$ chosen by Eve. However, at the same time, the transducer $\mathcal{H}$ is simulated by Adam on the composed BS-actions, and this output is analyzed (see below). Moreover, all branching choices are made by Adam. Thus, each play of $\mathcal{B}$ describes a branch of the binary tree, a run of $\mathcal{U}$ and a run of $\mathcal{U}'$ on this branch, and Adam's choices in the simulation of $\mathcal{H}$ on the output from these runs. The reason why Adam is in charge of the non-deterministic choices of the transducer $\mathcal{H}$ is that these choices aim at maximizing the S-values. Indeed, in $\mathcal{H}$, the output B-value is always equivalent to the input one, in any run of the automaton. Since maximizing the S-values is the job of Adam (he also controls the choices of the non-deterministic S-automaton $\mathcal{U}'$), he is in control of the transducer $\mathcal{H}$.

A subpath π of a play is called a *block* if the following hold:

- both a state from $F_B^{\mathcal{U}}$ and a state from $F_S^{\mathcal{U}'}$ occur during π.
- for every counter $\gamma\in\Gamma_B^{\mathcal{H}}$, if there is an increment for γ in π, there is also a reset for the same γ in π.

Let $K = |\Gamma_B^{\mathcal{H}}|$.

We will call a *block of level k* a block which occurs in an $\{\varepsilon,\mathtt{r}\}$-sequence of counter $k+1$ (a block of level K is just a normal block).

Given a state $(q_{\mathcal{U}}, q_{\mathcal{U}'}, q_{\mathcal{H}}, (i,j,z))\in Q$, and $k\in[0,K]$, $i(k)\in[1,m]$ is the number of the current block of level k, while component $j(k)$ is ϵ if no $F_S^{\mathcal{U}'}$ state has been seen in this block, $\bot$ if $F_S^{\mathcal{U}'}$ has been seen but no $F_B^{\mathcal{U}}$ following it, and $\top$ if $F_B^{\mathcal{U}}$ has been seen following

$F_S^{\mathcal{U}'}$. Finally, $z(k) = \varepsilon$ if no increment or reset has been seen in the current block, `ic` if there was an increment but no reset, and `r` if there was a reset.

The state $(q_{\mathcal{U}}, q_{\mathcal{U}'}, q_{\mathcal{H}}, (i, j, z))$ is accepting if the last component verifies that for all k, $j(k) \neq \bot$, or if it is $q_\top$.

A block of level k can be closed when an $F_S^{\mathcal{U}'}$ and then an $F_B^{\mathcal{U}}$ have been seen (so $j(k) = \top$), and simultaneously, if an `ic` for counter k has been seen, then an `r` has also been seen (i.e. $z(k) \in \{\varepsilon, \mathtt{r}\}$).

For all $k \in [0, K]$, the automaton starts from $i(k) = 1$.

Otherwise, we define how $\mathcal{B}$ updates i, j, z in a deterministic way depending on the counter actions and accepting states coming from the choices of Eve and Adam.

Resetting a level k means that for all $k' \leq k$, $i(k')$ is set to $i(k+1)$, $j(k)$ to ϵ, and $z(k)$ to `r`. We recall that the counter actions comes from the transducer $\mathcal{H}$, and are hierarchical.

(1) If a $F_S^{\mathcal{U}'}$ is seen, all $j(k) = \epsilon$ are updated to $\bot$.
(2) If a $F_B^{\mathcal{U}}$ is seen, all $j(k) = \bot$ are updated to $\top$.
(3) If an action $\mathtt{r}_k$ is seen (with ε for $k' > k$ and `r` for $k' \leq k$), then the automaton sets $z(k)$ to `r` in all $k' \leq k$.
(4) If an action $\mathtt{ic}_k$ is seen (with ε for $k' > k$ and `r` for $k' < k$), then the automaton resets level $k-1$. Moreover, for all $k' \geq k$ such that $z(k') = \varepsilon$, $z(k')$ is set to `ic`.
(5) After updates (1) to (4), if there is a k with $j(k) = \top$ and $z(k) \in \{\varepsilon, \mathtt{r}\}$, then $i(k)$ is incremented, $j(k)$ is set to ϵ and $z(k)$ to ε. Moreover if there is a $k' < k$ such that $i(k')$ is now smaller than $i(k)$, then $i(k')$ is set to $i(k)$.
(6) If some $i(k)$ is supposed to increase by (5) but is already at m, then the automaton moves to $q_\top$, and continues to simulate $\mathcal{U}$ and $\mathcal{U}'$. It can never leave $q_\top$ so the play is accepting.

Notice that by update rule (5), if the current block of level $k+1$ has number i, the automaton starts counting blocks of level k at the same number i, since it suffices to witness m consecutive blocks, regardless of their levels.

It is straightforward to verify that according to these updates, all reachables states are in Q_{use}.

E.3 Quasi-weakness of $\mathcal{B}$

We now show that $\mathcal{B}$ is a quasi-weak automaton.

We consider a path π of a n-run of $\mathcal{B}$. We define a configuration $C = (i_C, v_C, j_C, z_C)$ of the automaton $\mathcal{B}$ to be an element of $E_n = ([1, m] \times [0, n] \times \{\epsilon, \bot, \top\} \times \{\varepsilon, \mathtt{ic}, \mathtt{r}\})^{K+1} \cup \{q_\top\}$.

The new element v_C stores the value of counter k in $v_C(k)$ for all $k \in [1, K]$. Since level 0 does not correspond to a counter, we fix the value of $v_C(0)$ to be always 0. For other components, the set of values $[0, n]$ is sufficient because π is a path in a n-run, so counter values never go above n.

We now analyze the evolution of C on the path π, according to the transformations (1) to (6).

An *alternation* is a switching between accepting and rejecting states (or vice-versa).

We show that if a subpath of π begins and ends at the same configuration C, then it is alternation-free. We call such a subpath a cycle.

We consider a cycle. If no $i(k)$ is modified during the cycle, the other elements can only increase (for $\epsilon < \bot < \top$ and $\varepsilon < \mathtt{ic} < \mathtt{r}$), according to the operations (1) to (6)), so the cycle contains only one configuration, and hence is alternation-free.

Otherwise let k be maximal such that $i(k)$ changes during the cycle. In order to return to the same configuration, the cycle must contain a reset for $i(k)$. Notice that (4) is the only operation that resets some $i(k)$, and it occurs when $\mathtt{ic}_{k'}$ is read with $k' > k$. We choose a k' such that $\mathtt{ic}_{k'}$ is seen during the cycle. According to operations (1) and (2), $j(k')$ can only increase during the cycle, since level k' is not reset. Since the configuration C is the same in the beginning and in the end, we get that $j(k')$ must not change during the whole cycle.

We consider the possible cases for $j(k')$. If $j(k')$ is $\bot$, then the whole cycle is non-accepting, so there is no alternation.

We know that $v(k')$ is incremented at some point by action $\mathtt{ic}_{k'}$, so counter k' has to be reset in the cycle in order to match the original value. Thus if the value of $j(k')$ during the cycle is $\top$, it means that we will have $z(k') = \mathtt{r}$ and $j(k') = \top$ at the end of the cycle. Operation (5) implies that $i(k')$ has to be incremented, which is absurd because $k' > k$ implies that $i(k')$ does not change during the cycle, by choice of k.

The only remaining case is $j(k') = \epsilon$ during the whole cycle. By definition of Q_{use}, for all $k'' \leq k'$, $j(k'') = \epsilon$ during the whole cycle. Moreover, for $k'' > k'$, $j(k'')$ can only increase (by choice of k'), so in order to return to the initial configuration, they cannot change at all. Hence the cycle is alternation-free, since acceptance is determined only by the j components.

Any cycle is alternation-free, so the number of alternations in an n-run is bounded by the number of elements of E_n, which is less than $((m+1)(n+1)9)^{K+1}+1$. Hence the length of an alternating chain in an n-run of $\mathcal{B}$ is bounded by $\alpha(n) = ((m+1)(n+1)9)^{K+1}+1$. We remind that K and m are fixed by $\mathcal{U}$ and $\mathcal{U}'$, and do not depend on the input tree.

We can conclude that $\mathcal{B}$ is a quasi-weak alternating automaton.

E.4 Correctness of $\mathcal{B}$

Formal proof of $[\![\mathcal{B}]\!]_B \preccurlyeq [\![\mathcal{U}]\!]_B$

We will in fact show $[\![\mathcal{B}]\!]_B \leq [\![\mathcal{U}]\!]_B$.

Let t be a tree and $M \in \mathbb{N}$ such that $[\![\mathcal{U}]\!]_B(t) = M < \infty$. We want to show $[\![\mathcal{B}]\!]_B(t) \leq M$. Let R be a B-accepting run of $\mathcal{U}$ on t of value M.

Let σ be the strategy of Eve in the game $(\mathcal{B}, t)$ that consists in playing from the run R. We have $val_B(\sigma) = val_B(R) = M$, since $\mathcal{B}$ just copies the B-action of $\mathcal{U}$.

It remains to show that the strategy $R_{\mathcal{B}}$ is actually winning. Assume for the sake of contradiction that there is a play π compatible with σ which stabilizes in non-accepting states.

By definition of rejecting states, there is a $k \in [0, m]$ with $j(k) = \bot$ in any state of this partition. Let k be maximal such that there is $j(k) = \bot$ infinitely often in π. We show that after some point, $j(k)$ stabilizes in $\bot$. If k is the highest counter, then level k is never reset, so $j(k)$ stabilizes in $\bot$. Otherwise, assume $j(k)$ does not stabilize in $\bot$. There must be infinitely many updates of $j(k)$ from ϵ to $\bot$, and infinitely many resets of level k. Let $k' > k$. By choice of k, $j(k')$ must stabilize in ϵ or $\top$ (otherwise, $j(k')$ is $\bot$ infinitely often, contradicting the choice of k). If $j(k')$ stabilizes in ϵ, every update of $j(k)$ from ϵ to $\bot$ is simultaneously done

on $j(k')$ so this is absurd. If $j(k')$ stabilizes in $\bot$, every reset of level k is either also a reset for level k' or caused by an action $\mathtt{r}$ for counter k'. In both cases it is absurd, since $j(k') = \top$ and $z(k') = \mathtt{r}$ causes an increment of i and sets $j(k')$ back to ϵ (or m is reached and the automaton moves to $q_\top$).

This shows that $j(k')$ stabilizes in $\bot$.

This is absurd, because there are infinitely many accepting states $F_B^{\mathcal{U}}$ of $\mathcal{U}$ on π, so operation (2) in the transition table of $\mathcal{B}$ implies that $j(k)$ cannot stabilize in $\bot$ on π.

Formal proof of $[\![\mathcal{U}]\!]_B \preccurlyeq [\![\mathcal{B}]\!]_B$

Remark that since $\mathcal{H}$ is history-deterministic, there is a monotonic family of strategies $\delta = (\delta_n)_{n\in\mathbb{N}}$ and an α such that $\mathcal{H}^\delta \approxeq_\alpha \mathcal{G}$, where $\mathcal{H}^\delta$ is restricted to the use of runs of $\mathcal{H}$ driven by some $\delta_n \in \delta$.

Assume for the sake of contradiction that $[\![\mathcal{U}]\!]_B \not\preccurlyeq [\![\mathcal{B}]\!]_B$, i.e. there exists a sequence $(t_n)_{n\in\mathbb{N}}$ such that $\{[\![\mathcal{B}]\!]_B(t_n), n \in \mathbb{N}\}$ is bounded by some $M \in \mathbb{N}$, but $\{[\![\mathcal{U}]\!]_B(t_n), n \in \mathbb{N}\}$ is unbounded. Note that since $[\![\mathcal{U}]\!]_B \approx [\![\mathcal{U}']\!]_S$, we have $\{[\![\mathcal{U}']\!]_S(t_n), n \in \mathbb{N}\}$ also unbounded.

Let $\mathcal{A}$ be the non-deterministic automaton $\mathcal{H} \circ (\mathcal{U} \times \mathcal{U}')$. That is,

- $Q_\mathcal{A} := Q_\mathcal{U} \times Q_{\mathcal{U}'} \times Q_\mathcal{H}$ and $q_0^\mathcal{A} := (q_0^\mathcal{U}, q_0^{\mathcal{U}'}, q_0^\mathcal{H})$,
- $\Gamma_B^\mathcal{A} := \Gamma_B^\mathcal{H}$ and $\Gamma_S^\mathcal{A} := \Gamma_S^\mathcal{H}$,
- $F_B^\mathcal{A} := F_B^\mathcal{U}$ and $F_S^\mathcal{A} := F_S^{\mathcal{U}'}$,
- $\Delta_\mathcal{A} = \Delta_\mathcal{H} \circ (\Delta_\mathcal{U} \times \Delta_{\mathcal{U}'})$.

Remark that by Theorem 4 and Lemma 12, $\mathcal{A} \approxeq \mathcal{U} \times \mathcal{U}'$.

Let $N := \beta_M(|Q_\mathcal{A}| + 1)$, with β_M coming from the BS-equivalence $\mathcal{H}^\delta \approxeq_\alpha \mathcal{G}$. We choose n such that $[\![\mathcal{U}']\!]_S(t_n) \geq N$. Let R' be an accepting run of $\mathcal{U}'$ on $t := t_n$ with $val_S(R') > N$.

Let σ be an accepting strategy for Eve, witnessing acceptance of $\mathcal{B}$ over t with $val_B(\sigma) \leq M$. We look in particular at the set P of plays compatible with σ where Adam chooses to play from the run R' and uses strategy $\delta_{|Q_\mathcal{A}|+1}$ to drive $\mathcal{H}$. Notice that the only remaining choice for Adam concerns the branching, so P describes a binary tree of possible plays, which agree on common prefixes.

We know that in order for $val_B(\sigma) \leq M$, Eve must play a run R of $\mathcal{U}$ such that $val_B(R) \leq M$ on every branch of P. Let π be a branch of the binary tree described by P.

We first show that the strategy $\delta_{|Q_\mathcal{A}|+1}$ describes an accepting run of $\mathcal{H}$ on the word $v = (out_B(R,\pi), out_S(R',\pi))$ where $out_B(R,\pi)$ (resp. $out_S(R',\pi)$) are the infinite words describing the actions output by run R (resp. R') on the branch π. From $\mathcal{H}^\delta \approxeq_\alpha \mathcal{G}$, we get that there is a run $R_\mathcal{H}$ of $\mathcal{H}$ on v driven by some δ_p with $val_B(R_\mathcal{H}) \leq \alpha(M)$ and $\beta_M(val_S(R_\mathcal{H})) \geq N = \beta_M(|Q_\mathcal{A}| + 1)$. We get that either $p = val_S(R_\mathcal{H}) \geq |Q_\mathcal{A}| + 1$, or $val_S(R_\mathcal{H}) = \infty$. In both cases, by monotonicity of δ, the run of $\mathcal{H}$ on v driven by $\delta_{|Q_\mathcal{A}|+1}$ is accepting.

This means that the every play of P represents an accepting run of $\mathcal{H}$ over v and outputs hierarchical BS-actions with S-value at least N and B-value at most M.

Next we show that $\mathcal{B}$ must witness m blocks on every play in P (which will be used to build a cost trap on R and R').

Let π be a play in P.

If the automaton $\mathcal{B}$ reaches the accepting sink state $q_\top$ on π, it means that for some level k, m blocks have been witnessed. We assume by contradiction that it is not the case. Let k be

minimal such that $i(k)$ does not change infinitely many times in π, so it stabilizes to some value (for instance it is always the case for $i(K)$, which only increases).

We know that $j(k)$ stabilizes to ϵ or $\top$. But ϵ is impossible, because R' is accepting, so any $F_S^{\mathcal{U}'}$ would make $j(k)$ change to $\bot$ by operation (1). Hence $j(k)$ stabilizes in $\top$. This means that there is a finite number of resets for counter k, otherwise $i(k)$ would be incremented by operation (5). Since the play has bounded value, we can conclude that the number of increments for any counter $k' \geq k$ on π is also finite. But $i(k-1)$ is supposed to change infinitely many times, and this can only be done by incrementing infinitely many times higher counters (operation (4)). This is absurd, so π must witness m consecutive blocks, and stabilize in $q_\top$.

Now we build a cost trap for the automaton $\mathcal{A} \cong \mathcal{U} \times \mathcal{U}'$. Recall that the transition function of $\mathcal{B}$ is defined such that Eve selects a run of $\mathcal{U}$, Adam selects a run of $\mathcal{U}'$, and Adam controls $\mathcal{H}$ which outputs the hierarchical BS-actions corresponding to these composed runs. Let $R_\mathcal{A}$ be the run of $\mathcal{A}$ on t corresponding to the tree of plays P (ignoring the components of the state dealing with the blocks). We have $val_B(R_\mathcal{A}) \leq M$ and $val_S(R_\mathcal{A}) > |Q_\mathcal{A}|$.

We showed that for any branch π of $R_\mathcal{A}$, state $q_\top$ is reached on π, so m consecutive blocks are witnessed along this branch.

If π is a branch of $R_\mathcal{A}$, for all $i \in [0, m]$, we take for e_i^π the last position of π where some counter $i(k)$ has value i.

By the definition of the transitions of $\mathcal{B}$ (operations (1) to (6)), the path $[e_i^\pi, e_{i+1}^\pi)$ is always a block.

It remains to show the nesting condition of the cost trap definition. Let π_1, π_2 be two branches of $R_\mathcal{A}$, sharing some prefix up to position y, and let $x < y$ be the first position where they disagree : $e_i^{\pi_1} = x$ but $e_i^{\pi_2} > x$.

By definition of the e_i^π's, we have that x is the last position on π_1 where some counter $i(k)$ has value i. Assume $e_i^{\pi_2} \leq y$, then $e_i^{\pi_2}$ is on π_1, but some counter $i(k)$ has value i, which is absurd since $e_i^{\pi_2} > x$. We can conclude that $e_i^{\pi_2} > y$: the nesting condition is satisfied.

We have completed the proof that $R_\mathcal{A}$ is a cost trap for $\mathcal{A}$. By Proposition 9, this implies $[\![\mathcal{U}']\!]_S \not\preccurlyeq [\![\mathcal{U}]\!]_B$, which is a contradiction.