Contextual Equivalence for Alternation and Urgency

Eren Keskin, Roland Meyer, and Sören van der Wall
TU Braunschweig
Email: e.keskin@tu-bs.de, roland.meyer@tu-bs.de, s.van-der-wall@tu-bs.de

***Abstract*—We propose a new programming model with support for alternation, imperfect information, and recursion. We model imperfect information with the novel programming construct of urgency annotations that decorate the (angelic and demonic) choice operators and control the order in which the choices have to be made. Our contribution is a study of the standard notions of contextual equivalence for urgency programs. Our first main result are fully abstract characterizations of these relations based on sound and complete axiomatizations. Our second main result is to settle their computability status. Notably, we show that the contextual preorder is $(2\mathsf{h}-1)$-EXPTIME-complete for programs of maximal urgency h when the regular observable is given as an input resp. PTIME-complete when the regular observable is fixed. Our findings imply new decidability results for hyper model checking, a prominent problem in security.**

Keywords: Alternation, imperfect information, contextual equivalence, full abstraction, axiomatization, complexity.
Funding: This work was supported by the DFG project EDS@SYN: Effective Denotational Semantics for Synthesis.

I. Introduction

We propose the new programming construct of urgency annotations for alternating (angelic and demonic) choices. Alternating choices are usually resolved in program order [1]. Urgency annotations decorate the choice operators with a natural number that defines when a choice has to be made: the higher the urgency, the sooner. The lower urgency choices remain unresolved in the program term until all higher urgency choices have been made. Urgency thus models imperfect information by an out-of-program-order execution of choices. Consider an example, where we underline the next choice:

$$(t \wedge^1 f).\underline{(l \vee^2 r)} \quad \to \quad \underline{(t \wedge^1 f).l} \quad \to \quad f.l \ .$$

The demonic choice $\wedge^1$ of urgency 1 is resolved only after the angelic choice $\vee^2$ of urgency 2, although it is written earlier in the program text. Intuitively, the demonic choice is able to react to the angelic choice, and indeed urgency programs have a semantics in terms of game arenas. Besides alternation and urgency, our programming model supports recursion.

The idea of urgency annotations goes back to the recent work on hyperproperties for expressing security requirements [2]. A hyperproperty P for a language L over an alphabet Σ takes the form of an alternation

$$\forall w_1 \in L.\ \exists w_2 \in L\ \ldots\ Q w_n \in L.\ w_1 \oplus \ldots \oplus w_n \in P \ .$$

Here, $\oplus$ is the convolution and P is a language over Σ^n. The standard way of model checking hyperproperties is by repeatedly composing the program behind L with itself [3], [4]. Urgencies implement the mechanism of self-composition in a new way, by executing choices out-of-program-order. The advantage is that the urgent choices are harmonized with the control flow, while the quantifiers in hyper model checking are resolved without synchronization. As we show, this harmony yields decidability results for analyzing urgency programs even in the presence of recursion, while hyper model checking is undecidable despite visibility assumptions [5]. In Appendix J, we argue that the harmony enforced by urgency programs is justified from an application perspective.

We found it important to understand the impact of the new urgency annotations on the program semantics. We define the semantics of urgency programs in the standard way as contextual equivalence [6]. The notion of contextual equivalence depends on the level of detail at which we intend to observe the program behavior. We consider two standard definitions [7]:

$$\begin{array}{lllll} p \simeq q & \text{if} & \forall O.\forall c[\bullet]. & c[p] \Downarrow O & \text{iff} \quad c[q] \Downarrow O \\ p \simeq_O q & \text{if} & \forall c[\bullet]. & c[p] \Downarrow O & \text{iff} \quad c[q] \Downarrow O \ . \end{array}$$

The former definition quantifies over observables, and this is what we call *contextual equivalence*. The latter fixes an observable, like termination or reaching an error, and we refer to it as *O-specialized contextual equivalence*. Due to the alternation in urgency programs, observing is defined in a game-theoretic way: $c[q] \Downarrow O$ means Eve has a winning strategy in the arena $[\![c[q]]\!]$ when $O \subseteq \Sigma^*$ is the objective.

Our first contribution are full abstraction results. We show that contextual equivalence and its specialized variant coincide with congruence relations that do not quantify over contexts or observables. The congruences are defined axiomatically, and one may also say we axiomatize the contextual equivalences. An important insight is that imperfect information distributes over perfect information. In the example,

$$\begin{aligned} (t \wedge^1 f).(l \vee^2 r) &\simeq (t \wedge^1 f).l \vee^2 (t \wedge^1 f).r \\ &\simeq (t.l \wedge^1 f.l) \vee^2 (t.r \wedge^1 f.r) \ . \end{aligned}$$

The full abstraction results rely on several ingredients. For soundness, we establish a context lemma limiting the set of contexts we have to consider in the proof. For completeness, we first show that programs can be brought into a normal form that eliminates recursion and orders choices according to their urgency, similar to the last term in the two equations above. Then we devise characteristic contexts that tell apart programs which are not related axiomatically. There is a grain of salt: for the specialized contextual equivalence, completeness needs

a side condition on the observable. It is liberal and met by the observables typically used [7].

Our second contribution is to settle the computability status of the two contextual equivalences and their preorder variants. The main finding is that the specialized contextual preorder is $(2\mathsf{h}-1)$-EXPTIME-complete for programs of maximal urgency h when the regular observable is given as an input resp. PTIME-complete when it is fixed. To circumvent the above side condition on the observable, the algorithm does not rely on the full abstraction result but iterates through contexts. To get the upper bound right, an important idea is to factorize the set of contexts. We equate contexts $c[\bullet]$ that have the same solution space: the same set of programs p so that $c[p] \Downarrow O$. The challenge is to handle the factorization algorithmically. We show how to represent solution spaces explicitly using the novel concept of characteristic terms.

The remainder of the work is structured as follows. After an introduction to urgency programs, we state the full abstraction results in Section III, followed by the proofs of soundness, normalization, and completeness. In Section VII, we state the decidability and complexity results, followed by the upper and the lower bound proofs. We discuss related work in Section X and conclude in Section XI. Details missing in the main text and further results can be found in the appendix.

II. Programming Model

Throughout the development, we fix a natural number $\mathsf{h} > 0$ for the maximal urgency in programs, a finite alphabet Σ of terminal symbols with typical elements a, b, c and a finite or infinite set of non-terminals N with elements A, B, C. Each non-terminal has a so-called defining program term given by the function $E : N \to \mathbf{T}$.

The set $\mathbf{T}$ of *program terms* (of urgency up to h over Σ, N) is defined by the grammar

$$p \quad ::= \quad a \mid \mathsf{skip} \mid \mathsf{err} \mid A \mid p.p \mid \bigvee^{\mathsf{u}} P \mid \bigwedge^{\mathsf{u}} P \ .$$

Terminal symbols represent program commands with visible behavior, skip is a command without visible effect, and err aborts the computation unsuccessfully. Non-terminals model recursive functions, we have concatenation, and angelic ($\vee^{\mathsf{u}}$) as well as demonic ($\wedge^{\mathsf{u}}$) choice of urgency $0 < \mathsf{u} \leq \mathsf{h}$. We use $\bigcirc^{\mathsf{u}}$ to mean $\vee^{\mathsf{u}}$ or $\wedge^{\mathsf{u}}$, infix notation for binary choices, and $\bigcirc^{\mathsf{u}} p$ for $\bigcirc^{\mathsf{u}}\{p\}$. An action is a term A or $\bigcirc^{\mathsf{u}} P$. Terms that contain actions are called active. Terms that do not contain actions are called passive. Passive terms are words over $\Sigma \cup \{\mathsf{skip}, \mathsf{err}\}$, and we also call them word terms $w \in \mathbf{W}$. To avoid brackets, we let concatenation bind stronger than choices. Choices range over non-empty but possibly infinite sets of terms. Infinitary syntax requires care to make sure set and game-theoretic concepts remain sound. We moved the corresponding lemmas to Appendix A and B to keep the presentation light. The motivation for infinitary syntax will be given at the end of the section. We lift the notion of urgency from choices to program terms and define $\mathsf{urg} : \mathbf{T} \to \mathbb{N}$ by

$$\mathsf{urg}(\mathsf{skip}) = \mathsf{urg}(\mathsf{err}) = \mathsf{urg}(a) = 0 \qquad \mathsf{urg}(A) = \mathsf{h}$$
$$\mathsf{urg}(p.q) = \max\{\mathsf{urg}(p), \mathsf{urg}(q)\} \qquad \mathsf{urg}(\bigcirc^{\mathsf{u}} P) = \mathsf{u} \ .$$

A *context* $c[\bullet]$ (of maximal urgency h over Σ and N) is a term that contains at most one occurrence of the fresh non-terminal $\bullet$. The set of all contexts is denoted by $\mathbf{C}$. The expression $c[p]$ refers to the term obtained from $c[\bullet]$ by replacing $\bullet$ with the term p. A term q is a subterm of p if there is a context $c[\bullet]$ with $p = c[q]$. A subterm is called outermost if it is not enclosed by a choice. For example, p is outermost in $p.q$ but not in $(p \vee^{\mathsf{u}} q).r$. The *leading subterm* $\mathsf{lead}(p)$ of a term p is defined as the outermost action with the highest urgency. If several outermost actions have this urgency, then the leftmost of them is leading. Passive terms do not have leading subterms. Where helpful, we will underline a subterm that contains the leading subterm. Note that $\underline{p.q}$ implies $\mathsf{urg}(p) \geq \mathsf{urg}(q)$ and $p.\underline{q}$ implies $\mathsf{urg}(q) > \mathsf{urg}(p)$. We denote the unique context enclosing the leading subterm $\mathsf{lead}(p)$ in p by $\mathsf{en}_p[\bullet] \in \mathbf{C}$. We have $p = \mathsf{en}_p[\mathsf{lead}(p)]$.

```
if (x < y) { r = mem[x]; /* now do work on r */ }
```

Listing 1: Branch prediction.

Consider the program in Listing 1. We present two models for it that explain the influence of branch prediction on the program semantics. The first is the term $\underline{(t \wedge^1 f).(l \vee^1 r)}$. It captures the program semantics without branch prediction. The terminals $t, f \in \Sigma$ stand for the outcome of $\mathtt{x} < \mathtt{y}$. The terminals $l, r \in \Sigma$ stand for the corresponding branches. The choice $(t \wedge^1 f)$ is demonic (or owned by player Adam), because the program has no influence on the outcome of the condition. The choice $(l \vee^1 r)$ is angelic (or owned by Eve), because it is up to the program and its execution environment to execute a branch. Since the urgencies coincide, the choices are made from left to right. Eve can thus see the outcome of the condition and select the correct branch.

The second model is $(t \wedge^1 f).\underline{(l \vee^2 r)}$ from the introduction. Given the higher urgency of the angelic choice, Eve is forced to decide on a branch before seeing the outcome of the condition. Adam can thus enforce a branch prediction error.

The latter example is the essence of the Spectre attack [8]. The attacker trains the processor to speculate on a branch guarded by a condition on pointer `x`, with an address held by `x` that does not satisfy the guard. The speculation then loads information from this unexpected address, and the information remains in the cache even though the speculation is undone.

A. Semantics

Given that our programming model has alternation, the operational semantics of a term is not a plain transition system but a *game arena* $[\![p]\!] = (\mathbf{T}, p, \mathsf{own}, \to)$ in which positions are owned by player *Eve* or player *Adam*. The set of positions is the set of all terms. The initial position is the given term. The ownership assignment $\mathsf{own} : \mathbf{T} \to \{Eve, Adam\}$ returns

the owner of the leading subterm, $\mathsf{own}(p) = \mathsf{own}(\mathsf{lead}(p))$. Adam owns the demonic choices, $\mathsf{own}\left(\bigwedge^{\mathsf{u}} P\right) = \mathit{Adam}$, and Eve owns the angelic choices, $\mathsf{own}\left(\bigvee^{\mathsf{u}} P\right) = \mathit{Eve}$. We also give skip, err, terminals, and non-terminals to Eve. This has no influence on the semantics as there will be at most one move from these positions. The set of moves is defined as the smallest relation satisfying the following rules:

$$\frac{p \in P}{\bigcirc^{\mathsf{u}} P \to p} \qquad \frac{}{A \to E(A)} \qquad \frac{\mathsf{lead}(p) \to q}{p \to \mathsf{en}_p[q]} \ .$$

A move always rewrites the leading subterm. For a choice, it selects one alternative. For a non-terminal, it inserts the defining term. We define $\mathsf{succ}(p) = \{q \mid p \to q\}$. It is worth noting that the game arena has perfect information. Imperfect information is modeled through choices, and they are eventually resolved.

The operational semantics is intensional in that it gives precise information about the program state at runtime. From a programming perspective, what matters is the result of a computation or, more generally, the *observable behavior* of the program. Due to Adam's influence, the observable behavior will rarely be a single word but rather a language $O \subseteq \Sigma^*$. We write $p \Downarrow O$ to mean that Eve can enforce termination and the result will be a word from O, no matter how Adam plays. We make this precise.

Our notion of observable behavior is based on concepts from game theory. We refer to a language $O \subseteq \Sigma^*$ as a reachability objective for the game arena $[\![p]\!]$. A play in this arena is a maximal (finite or infinite) sequence of positions $\pi = p_0, p_1, \ldots$ that starts in the given term, $p_0 = p$, and respects the moves of the game arena, $p_i \to p_{i+1}$ for all i. If the play ends, the result is a word term $w \in \mathbf{W}$. We interpret it as an element of the monoid with zero $(\Sigma^* \cup \{\mathsf{err}\}, . \, , \mathsf{skip}, \mathsf{err})$. Here, skip is the unit, often denoted by ε, and err is the zero. We use $\stackrel{*}{=}$ to denote the monoid equality. Eve wins the play when $w \in O$, meaning there is $v \in O$ so that $w \stackrel{*}{=} v$. Otherwise, Adam wins the play. In particular, Adam wins all infinite plays and all plays exhibiting $\mathsf{err} \notin \Sigma$.

A positional strategy for Eve is a function $\sigma : \mathbf{T} \to \mathbf{T}$ so that $q \to \sigma(q)$ for all terms q owned by Eve that admit further rewriting. Since we are interested in reachability objectives, we can use positional strategies without loss of generality [9]. A play π is conform to σ if for all i with $\mathsf{own}(p_i) = \mathit{Eve}$ and $\mathsf{succ}(p_i) \neq \emptyset$ we have $\sigma(p_i) = p_{i+1}$. Eve wins objective O, if she has a strategy σ so as to win all plays that are conform to this strategy. This is what we denote by $p \Downarrow O$.

For the branch prediction example, the operational semantics of $(t \wedge^1 f).(l \vee^1 r)$ and $(t \wedge^1 f).(l \vee^2 r)$ is given in Figure 1. Rectangular nodes are owned by Adam, circular nodes by Eve. We also use rectangular nodes with rounded corners where the ownership does not matter. The objective for correct branch prediction is $O = \{t.l, f.r\}$. We have $(t \wedge^1 f).(l \vee^1 r) \Downarrow O$ but $(t \wedge^1 f).(l \vee^2 r) \not\Downarrow O$.

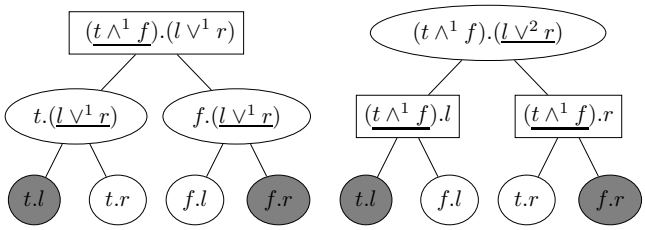


Fig. 1: The game arenas $[\![(t \wedge^1 f).(l \vee^1 r)]\!]$ and $[\![(t \wedge^1 f).(l \vee^2 r)]\!]$. The objective is given in gray.

B. Contextual Preorder

The notion of observable behavior is not compositional: we may have $p \Downarrow O$ if and only if $q \Downarrow O$ for all objectives O, yet the two terms behave differently when placed into a context. In our example, $(l \vee^1 r) \Downarrow U$ if and only if $(l \vee^2 r) \Downarrow U$ for all objectives $U \subseteq \Sigma^*$. When inserting the terms into the context $c[\bullet] = (t \wedge^1 f).\bullet$, however, we have the difference discussed above. This is a common problem, and the way out is to consider the largest congruence that lives inside observational equivalence. It is more elegant to work with a precongruence and define the congruence of interest as a derived notion.

Definition 1: The *contextual preorder* $\preceq \subseteq \mathbf{T} \times \mathbf{T}$ between urgency terms is defined by $p \preceq q$, if

$$\forall c[\bullet] \in \mathbf{C}.\ \forall O \subseteq \Sigma^*.\ c[p] \Downarrow O \text{ implies } c[q] \Downarrow O \ .$$

Fix an objective $O \subseteq \Sigma^*$. The *O-specialized contextual preorder* $\preceq_O \subseteq \mathbf{T} \times \mathbf{T}$ is defined by $p \preceq_O q$, if

$$\forall c[\bullet] \in \mathbf{C}.\ c[p] \Downarrow O \text{ implies } c[q] \Downarrow O \ .$$

The *contextual equivalence* is then $\simeq \ = \ \preceq \cap \succeq$, and the *$O$-specialized contextual equivalence* is $\simeq_O = \preceq_O \cap \succeq_O$.

In the example, we have $(l \vee^2 r) \preceq (l \vee^1 r)$ and so by congruence $(t \wedge^1 f).(l \vee^2 r) \preceq (t \wedge^1 f).(l \vee^1 r)$. The reverse does not hold, consider context $\bullet$ and objective $\{t.l, f.r\}$ from above. Note that $p \preceq q$ implies $p \preceq_O q$ for every O.

With $O_{Term} = \Sigma^*$ as the objective, we can use the specialized contextual equivalence to study the termination behavior of programs. We can also introduce a letter loc so that $p \Downarrow O_{Reach}$ with $O_{Reach} = \Sigma^*.\mathsf{loc}.\Sigma^*$ observes visits to a specific location. Contextual equivalence is more precise and takes into account all objectives. Both notions are also motivated by verification, where contextual equivalence gives information about which information can be abstracted away from an urgency term without an influence on the objective, similar to how bisimilarity preserves CTL* properties [10].

The motivation for an infinite set of non-terminals and infinitary terms is to model parameterized functions in a simple yet general way. The idea is to introduce a non-terminal for each instantiation of the function's formal parameters by actual values, inspired by value passing in process algebra [11].

III. Full Abstraction

We define a precongruence $\sqsubseteq \subseteq \mathbf{T} \times \mathbf{T}$ on program terms that neither quantifies over contexts nor objectives but relates

terms solely based on their syntactic structure. The relation is defined through a set of axioms that should be understood as explaining the interplay between the operators in our programming model. The main finding is that this axiomatic precongruence coincides with the contextual preorder, and we also say that we axiomatize (in a sound and complete way) the contextual preorder. This is our main theorem.

Theorem 1 (Full Abstraction 1): $p \sqsubseteq q$ if and only if $p \preceq q$.

We also have a corresponding result for the O-specialized contextual preorder. In this setting, a complete axiomatization is considerably more difficult to obtain because, intuitively, we have to understand the concatenation behavior of language O. Our solution is partial in that we impose a side condition on the objective: it should be right-separating, a notion we will define in a moment when we have more technicalities at hand. It is always sound to reason with the precongruence.

Theorem 2 (Full Abstraction 2): $p \sqsubseteq_O q$ implies $p \preceq_O q$. If O is right-separating, then also $p \preceq_O q$ implies $p \sqsubseteq_O q$.

Luckily, the objectives of interest are right-separating.

Lemma 1: O_{Term} and O_{Reach} are right-separating.

We understand the concatenation behavior of an objective with the help of a syntactic precongruence relation [12] over the monoid $\Sigma^* \cup \{\mathsf{err}\}$. It may relate terminal words to err in case they cannot be extended to a word from the objective.

Definition 2: The *syntactic precongruence* induced by O on $\Sigma^* \cup \{\mathsf{err}\}$ is defined by $w \leq_O^s v$, if

$$\forall x, y \in \Sigma^*.\ x.w.y \in O \text{ implies } x.v.y \in O\ .$$

An objective is then right-separating, if the concatenation from left in the above definition is not needed to distinguish words. Formally, we define $\leq_O^r$ on $\Sigma^* \cup \{\mathsf{err}\}$ by $w \leq_O^r v$, if

$$\forall y \in \Sigma^*.\ w.y \in O \text{ implies } v.y \in O\ .$$

Definition 3: Objective O is *right-separating*, if $\leq_O^s = \leq_O^r$.

The syntactic congruence $\cong_O^s = \leq_O^s \cap \geq_O^s$ induced by the objective $O = \{t.l, f.r\}$ from the example has the classes $\Sigma^* \cup \{\mathsf{err}\}/_{\cong_O^s} = \{\{\mathsf{skip}\}, \{t\}, \{f\}, \{l\}, \{r\}, \{t.l, f.r\}\}$ plus a class for the remaining words.

Intuitively, right-separating objectives allow us to evaluate the O-specialized contextual preorder by using contexts $\bullet.r$ that only append to the right. For arbitrary objectives, we have to consider contexts $s.\bullet.r$ and it is difficult to understand the interplay between high urgencies in r and low urgencies in s.

We now give the two axiomatizations and explain them on an intuitive level. Recall that a precongruence is a reflexive and transitive relation that is preserved when inserting related terms into the same context.

Definition 4: The *axiomatic precongruence* $\sqsubseteq \subseteq \mathbf{T} \times \mathbf{T}$ is the least precongruence satisfying the axioms in Figure 2 except (S). The *O-specialized axiomatic precongruence* $\sqsubseteq_O$ on terms is the least precongruence satisfying all axioms in Figure 2. We use $\equiv$ for $\sqsubseteq \cap \sqsupseteq$ and $\equiv_O$ for $\sqsubseteq_O \cap \sqsupseteq_O$.

With the axioms given in Figure 2a, the choice operators span a completely distributive lattice on each urgency. The monotonicity axiom (L1) is not covered by the precongruence but implements an infinite replacement. The axiom has a side condition that can be found Appendix A. Due to this axiom, nodes in our proof trees may have an infinite degree. Yet, every path is guaranteed to be finite. To see the premise in axiom (L5), consider $p = (t \wedge^2 f)$ and context $\bullet.(l \vee^2 r)$. Then in $p.(l \vee^2 r)$ Eve wins while in $(p \vee^1 p).(l \vee^2 r)$ she loses, similar to Figure 1. For (L3), the reasoning is similar. As a consequence of the lattice axioms, one can derive the dual rules of (L2) and (L5). Distributivity (L2) states that the order of choices can be changed by considering all choice functions $f : I \to \bigcup_{i \in I} P_i$ with $f(i) \in P_i$ for all $i \in I$, denoted by $f : I \to P_I$.

The distributivity in (D1) captures the essence of imperfect information: concatenation to the left distributes over choice, provided the internals of the term are invisible because the choice has a higher urgency. The distributivity from the right in (D2) is similar but takes into account that the leading subterm for equal urgencies is leftmost. This clean interplay between imperfect information and choice came as a surprise

(a) Lattice

$$\text{(L1)}\ \frac{\forall i \in I.\ p_i \sqsubseteq q_i}{\bigcirc^{\mathsf{u}}\{p_i \mid i \in I\} \sqsubseteq \bigcirc^{\mathsf{u}}\{q_i \mid i \in I\}}$$

$$\text{(L2)}\ \frac{}{\bigvee^{\mathsf{u}}_{i \in I} \bigwedge^{\mathsf{u}} P_i \equiv \bigwedge^{\mathsf{u}}_{f : I \to P_I} \bigvee^{\mathsf{u}}\{f(i) \mid i \in I\}}$$

$$\text{(L3)}\ \frac{\mathsf{urg}(p) \leq \mathsf{u}}{p \wedge^{\mathsf{u}} (p \vee^{\mathsf{u}} q) \equiv p \qquad p \vee^{\mathsf{u}} (p \wedge^{\mathsf{u}} q) \equiv p}$$

$$\text{(L4)}\ \frac{}{\bigcirc^{\mathsf{u}}_{i \in I} \bigcirc^{\mathsf{u}} P_i \equiv \bigcirc^{\mathsf{u}} \bigcup_{i \in I} P_i} \qquad \text{(L5)}\ \frac{\mathsf{urg}(p) \leq \mathsf{u}}{p \sqsubseteq p \vee^{\mathsf{u}} q}$$

(b) Distributivity

$$\text{(D1)}\ \frac{\mathsf{urg}(p) < \mathsf{u}}{p.(\bigcirc^{\mathsf{u}} Q) \equiv \bigcirc^{\mathsf{u}}\{p.q \mid q \in Q\}}$$

$$\text{(D2)}\ \frac{\mathsf{urg}(p) \leq \mathsf{u}}{(\bigcirc^{\mathsf{u}} Q).p \equiv \bigcirc^{\mathsf{u}}\{q.p \mid q \in Q\}}$$

(c) Least element

$$\text{(B1)}\ \frac{}{\mathsf{err} \sqsubseteq p}$$

(d) Specialization

$$\text{(S)}\ \frac{w \leq_O^s v}{w \sqsubseteq_O v}$$

(e) Normalization

$$\text{(N)}\ \frac{\mathsf{v} < \mathsf{u}}{\bigvee^{\mathsf{v}} \bigcirc^{\mathsf{u}} Q \equiv \bigvee^{\mathsf{v}} \bigcirc^{\mathsf{v}} Q}$$

(f) Monoid

$$\text{(M)}\ \frac{w \stackrel{*}{=} v}{w \equiv v}$$

(g) Fixed point

$$\text{(FP)}\ \frac{}{A \equiv E(A)} \qquad \text{(LFP)}\ \frac{\forall A \in N.\ E(A)\{N/p_N\} \sqsubseteq p_A}{B \sqsubseteq p_B}$$

Fig. 2: Axioms defining $\sqsubseteq$ and $\sqsubseteq_O$.

to us and we consider these laws an important contribution.

A string with $\mathsf{err} \notin \Sigma$ is the most disadvantegous term for Eve, because it belongs to no objective $O \subseteq \Sigma^*$.

The monoid axiom (M) refers to word terms $w, v \in \mathbf{W}$. We interprete them in the monoid $\Sigma^* \cup \{\mathsf{err}\}$ and inherit the equality there, denoted by $\stackrel{*}{=}$ above. The equality strips brackets and skip, and interprets err as zero.

The normalization axiom (N) reflects the fact that only the outermost choice operator determines the urgency of a term. Towards soundess, note that once the outer choice with urgency u is resolved, we are sure that the context to the left has urgency strictly smaller than u and the context to the right has urgency at most u. Hence, the inner choice is the next to be resolved, independent of whether its urgency is u or $\mathsf{v} \geq \mathsf{u}$.

The fixed-point axiom (FP) allows us to rewrite non-terminals to their defining terms. The axiom (LFP) allows us to rewrite non-terminals to a prefixed point, using Knaster & Tarski's characterization of least fixed points as least prefixed points [13]. Here, we let p_N denote a vector of terms with one entry p_A per non-terminal $A \in N$, and use $\{N/p_N\}$ for the substitution of all non-terminals by these terms.

Recall that Axiom (S) only plays a role in the definition of the specialized axiomatic congruence, and that it depends on the objective O of interest (this is a family of axioms). The axiom relates word terms $w, v \in \mathbf{W}$ as prescribed by the syntactic precongruence. It thus incorporates the equality in the monoid $\Sigma^* \cup \{\mathsf{err}\}$, similar to axiom (M).

We state two more useful proof rules, which follow from the axioms, Appendix G:

$$\text{(REP)}\ \frac{\forall A \in N.\ p_A \sqsubseteq A}{q\{N/p_N\} \sqsubseteq q} \qquad \text{(L6)}\ \frac{\mathsf{urg}(p) \leq \mathsf{u}}{p \equiv \bigcirc^{\mathsf{u}} p}\ .$$

In the running example, we claimed that $(l \vee^2 r) \preceq (l \vee^1 r)$. We now prove this axiomatically and derive

$$l \vee^2 r \overset{\text{(L5)}}{\sqsubseteq} (l \vee^1 r) \vee^2 (l \vee^1 r) = \bigvee^2 (l \vee^1 r) \overset{\text{(L6)}}{\equiv} l \vee^1 r\ .$$

By (L5), we have $l \sqsubseteq l \vee^1 r$ and $r \sqsubseteq l \vee^1 r$, which we can apply to subterms due to congruence. The equality holds because choices range over sets.

The next sections are devoted to proving the axiomatization sound (Section IV) and, with the help of a normal form result (Section V), complete (Section VI).

IV. Soundness

Soundness is the left-to-right implication in Theorem 1 and Theorem 2: the axiomatic precongruence implies the contextual preorder, and likewise for the specialized case:

Proposition 1 (Soundness): $p \sqsubseteq q$ implies $p \preceq q$ and $p \sqsubseteq_O q$ implies $p \preceq_O q$.

This section is devoted to proving Proposition 1. Proving soundness is difficult because it requires us to reason over all contexts. If $p \sqsubseteq_O q$ is an axiom, then we need to show that $c[p] \Downarrow O$ implies $c[q] \Downarrow O$ for all $c[\bullet] \in \mathbf{C}$. We first develop a proof technique for soundness that allows us to reduce the set of contexts we have to consider, and in a second step prove the axioms sound.

A. Proof Technique

To define the contexts that have to be considered, we introduce some terminology. We say that term p is *immediate* for context $c[\bullet] \in \mathbf{C}$, if $c[p] \in \mathbf{W}$ or $c[p]$ is active and p contains the leading subterm in $c[p]$, denoted by $c[\underline{p}]$. If this is not the case, we call p *paused* for $c[\bullet]$. As the names suggest, immediate terms get rewritten in the next move while paused terms do not. To give an example, term $\bigvee^2 P$ is immediate for context $\bullet$. $\bigwedge^1 Q$ but paused for contexts $\bullet$. $\bigwedge^3 Q$ and $\bigvee^2 \{\bullet, q\}$. In the last context, $\bullet$ is enclosed by a choice. No term will be immediate for such a context.

To prove axioms $p \sqsubseteq_O q$ sound, we rely on this lemma.

Lemma 2 (Proof Technique): If $c[p] \Downarrow O$ implies $c[q] \Downarrow O$ for all contexts $c[\bullet] \in \mathbf{C}$ where at least one of p or q is immediate, then $p \preceq_O q$.

The proof of Lemma 2 relies on an observation: a paused term does not change the outcome of a move in the context.

Lemma 3: Consider terms p and q that are paused for context $c[\bullet]$. Then, $\mathsf{own}(c[p]) = \mathsf{own}(c[q])$ and there is a set of contexts $D \subseteq \mathbf{C}$ so that $\mathsf{succ}(c[p]) = \{d[p] \mid d[\bullet] \in D\}$ and $\mathsf{succ}(c[q]) = \{d[q] \mid d[\bullet] \in D\}$.

Proof Sketch: If a term is paused for a context, then one of the following will hold: the context variable $\bullet$ is enclosed by a choice, the term has a lower urgency than the leading term in the context, or the urgencies coincide but the context variable is placed to the right of the context's leading term. Therefore, both terms must be owned by the same player. The argumentation also shows that the moves result in similar (equal up to the inserted term) sets of successors. ∎

Proof of Lemma 2: Assume Eve wins O from $c[p]$. Then she does so in at most β-many moves, where β is an ordinal that is guaranteed to exist by results in Appendix B. To be clear, even for transfinite β, Eve wins each play after a finite number of moves. The ordinal β limits the size of the game arena reachable from $c[p]$ when she plays according to her strategy. We show $c[q] \Downarrow O$ by transfinite induction on β. The base case is simple, yet instructive. If $\beta = 0$ then $c[p] \in O$, meaning $c[p] \in \mathbf{W}$. Then p is immediate for context $c[\bullet]$, and the premise yields $c[q] \Downarrow O$.

In the inductive case, it will make no difference whether β is a limit ordinal or a successor ordinal, so we will not distinguish the two. We have $c[p] \Downarrow O$. If p or q is immediate for $c[\bullet]$, then the premise of the lemma already tells us $c[q] \Downarrow O$. Therefore, assume both terms are paused for $c[\bullet]$. Intuitively, we will see that Eve can copy her strategy from $c[p]$ to $c[q]$ (until the inserted term becomes immediate). Since both terms are paused for $c[\bullet]$, we can apply Lemma 3. It shows that, after insertion, the owner is the same, $\mathsf{own}(c[p]) = \mathsf{own}(c[q])$, and also gives a set of contexts $D \subseteq \mathbf{C}$ capturing the successors. Let $\mathsf{own}(c[p]) = \mathsf{own}(c[q]) = \mathit{Eve}$. Then, there must be a context $d[\bullet] \in D$ so that Eve wins $d[p]$ in $\beta' < \beta$ moves. By the induction hypothesis, $d[q] \Downarrow O$. Moreover, Eve can play $c[q] \to d[q]$ and win. If $\mathsf{own}(c[p]) = \mathsf{own}(c[q]) = \mathit{Adam}$,

then for all $d[\bullet] \in D$, Eve must win $d[p]$ in $\beta' < \beta$ turns. By the induction hypothesis, we get $d[q] \Downarrow O$ for all $d[\bullet] \in D$. This is exactly $\mathsf{succ}(c[q])$, so we have $c[q] \Downarrow O$ as well. ■

B. Soundness Proof

We are now prepared to prove the axioms sound. Using Lemma 2, all proofs share a common approach: we fix an objective and pick a context that is immediate for at least one term in the axiom. Then, we unroll the game arena for a few moves until it reveals the winning implication we are after. We restrict our attention to the more insightful arguments and defer the details to the appendix.

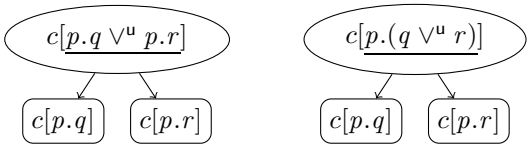
The proofs make use of the two properties of immediate terms given in the next lemma. They immediately follow from the definition of the successor relation.

Lemma 4: Let term p be immediate for context $c[\bullet]$. Then we have $\mathsf{own}\,(c[p]) = \mathsf{own}\,(p)$. If $\mathsf{urg}(p') \geq \mathsf{urg}(p)$, then also term p' is immediate for $c[\bullet]$.

Proof of Proposition 1 (Soundness): **Axiom (D1):** We consider binary choices owned by Eve, the generalization to choices over arbitrary sets and also Adam's case are similar. For $\mathsf{urg}(p) < \mathsf{u}$, the axiom says that $p.(q \vee^{\mathsf{u}} r) \sqsubseteq p.q \vee^{\mathsf{u}} p.r$ and vice versa. The goal is thus to show

$$p.(q \vee^{\mathsf{u}} r) \quad \simeq \quad p.q \vee^{\mathsf{u}} p.r.$$

Consider an objective O and let $c[\bullet]$ be a context for which at least one of $p.(q \vee^{\mathsf{u}} r)$ or $p.q \vee^{\mathsf{u}} p.r$ is immediate. The urgencies of both terms are u. This means that not only one but actually both terms are immediate for $c[\bullet]$ and, moreover, the owner of $c[\underline{p.(q \vee^{\mathsf{u}} r)}]$ and $c[\underline{p.q \vee^{\mathsf{u}} p.r}]$ is Eve, Lemma 4. The first moves in the game arenas are thus done by the same player and have the same result:



As a consequence, it is straightforward to translate winning strategies between the game arenas and we get the equivalence $c[p.(q \vee^{\mathsf{u}} r)] \Downarrow O$ if and only if $c[p.q \vee^{\mathsf{u}} p.r] \Downarrow O$.

Axiom (LFP): The proof makes heavy use of substitution. For the sake of readability, we use q^N for $q\{N/p_N\}$. We also extend this notation to contexts and write $d^N[\bullet]$ for the context obtained from $d[\bullet] \in \mathbf{C}$ by replacing all non-terminals $A \in N$ (different from $\bullet$) by p_A.

Assuming $E(A)^N \sqsubseteq p_A$ holds for all $A \in N$, the axiom yields $B \sqsubseteq p_B$. We proceed by an (outer) induction on the ordinal height of proof trees. The induction hypothesis yields $E(A)^N \preceq p_A$ for all $A \in N$. We have to show $B \preceq p_B$. Consider an objective O and a context $d[\bullet]$ for which p_B or B is immediate. We prove that $d[B] \Downarrow O$ implies $d[p_B] \Downarrow O$ with a detour. If Eve wins O from B, she does so in β-many moves, with β an ordinal, Appendix B. We apply transfinite induction on β to establish the following more general statement. For all contexts $c[\bullet]$ and all terms p, if Eve wins O from $c[p]$ in β moves, then she wins O from $c[p^N]$. Letting $c[\bullet] = d[\bullet]$ and $p = B$ gives us the desired conclusion.

The base case $\beta = 0$ is trivial: the term $c[p]$ must be a word term and hence $c[p] = c[p^N]$. Before moving on with the inductive step, we make a remark. Since the urgency of non-terminals is maximal, we have $\mathsf{urg}(A) \geq \mathsf{urg}(p)$ for all non-terminals A and all terms p. Moreover, the urgency of a term is monotonic in the urgency of its subterms, so in particular $\mathsf{urg}(q) \geq \mathsf{urg}(q^N)$ holds.

In the inductive step, let p be a term and $c[\bullet]$ be a context so that Eve wins O from $c[p]$ in β moves. We first consider the case that p is paused for $c[\bullet]$. Since $\mathsf{urg}(p) \geq \mathsf{urg}(p^N)$, Lemma 4 tells us that p^N is also paused for $c[\bullet]$. Similar to the proof of Lemma 2, we can use the induction hypothesis to argue that Eve wins O from $c[p^N]$. It is worth noting that in the paused case the move will change the surrounding context, which is why we strengthened the inductive statement to universally quantify over contexts.

Assume p is immediate for $c[\bullet]$. Let $q = \mathsf{lead}(p)$ be the leading subterm and recall that $p = \mathsf{en}_p[q]$, the term can be written as the unique context enclosing the leading subterm with the leading subterm inserted. The substitution distributes to all subterms and we also have $p^N = \mathsf{en}(p)^N[q^N]$. Showing that Eve wins from $c[p^N]$ thus means to show that she wins from $c[\mathsf{en}(p)^N[q^N]]$. Since q is leading in p, it is a choice or a non-terminal. We begin with the choice, $q = \bigcirc^{\mathsf{u}} Q$. We argue that q^N must be immediate for $c[\mathsf{en}(p)^N[\bullet]]$. To see this, note that p is immediate for $c[\bullet]$ and so q is immediate for $c[\mathsf{en}_p[\bullet]]$. The substitution distributes over the choice and we have $(\bigcirc^{\mathsf{u}} Q)^N = \bigcirc^{\mathsf{u}}\{r^N \mid r \in Q\}$. This shows $\mathsf{urg}(q) = \mathsf{urg}(q^N)$. For the outermost actions in $c[\mathsf{en}_p[\bullet]]$, the substitution can only lower the urgencies.

The fact that q^N is immediate for $c[\mathsf{en}(p)^N[\bullet]]$ yields

$$\begin{aligned}\mathsf{succ}(c[\mathsf{en}(p)^N[\underline{q^N}]]) &= \{c[\mathsf{en}(p)^N[r^N]] \mid q \to r\} \\ &= \{c[s^N] \mid p \to s\}.\end{aligned}$$

We also have $\mathsf{succ}(c[\underline{p}]) = \{c[s] \mid p \to s\}$. Similar to the proof of Lemma 2, Eve wins from $c[s]$ in $\beta' < \beta$ turns for all/one $s \in \mathsf{succ}(p)$, depending on the owner of $c[p]$. For every successor s, the induction hypothesis tells us that $c[s] \Downarrow O$ implies $c[s^N] \Downarrow O$. Since the owner of both $c[p]$ and $c[p^N]$ is the owner of the choice $\bigcirc$, Eve can copy her strategy.

It remains to consider the case that q is a non-terminal A. Then, $c[\mathsf{en}_p[\underline{A}]]$ can only be played into $c[\mathsf{en}_p[E(A)]]$ and Eve wins from this position in $\beta' < \beta$ moves. By the induction hypothesis, $c[\mathsf{en}_p[E(A)]^N]$ is also won by Eve. We can write this term as $c[\mathsf{en}(p)^N[\underline{E(A)^N}]]$. The hypothesis of the outer induction yields $E(A)^N \preceq p_A$. Therefore, Eve must also win from $c[\mathsf{en}(p)^N[p_A]] = c[\mathsf{en}_p[A]^N] = c[p^N]$. ■

V. Normalization

As a first step towards proving completeness, we now show that each term can be brought into a normal form using the axiomatic congruence. Normalization is a standard approach in

completeness proofs. New here is the treatement of alternation and urgency.

The normal form eliminates non-terminals and orders the interplay between concatenation and choice: a normal form term is a tree of height 2h (with h the maximal urgency) that repeatedly alternates between Eve's and Adam's choices while decreasing the urgency. The leaves of the tree are terminal words, skip, or err. We define the normal form terms by induction on the urgency: $NF_0 = \Sigma^* \cup \{\mathsf{err}\}$ and for $\mathsf{u} > 0$

$$\begin{aligned} ANF_{\mathsf{u}} &= \{\bigwedge^{\mathsf{u}} P \mid \emptyset \neq P \subseteq NF_{\mathsf{u}-1}\} \\ NF_{\mathsf{u}} &= \{\bigvee^{\mathsf{u}} P \mid \emptyset \neq P \subseteq ANF_{\mathsf{u}}\} \ . \end{aligned}$$

The base case terms are all owned by Eve. In an ANF_{u} term, Adam chooses over $NF_{\mathsf{u}-1}$ terms. In an NF_{u} term, Eve chooses over such ANF_{u} terms owned by Adam. The main result of this section is the following.

Proposition 2: There is a function $\mathsf{nf} : \mathbf{T} \to NF_{\mathsf{h}}$ so that for all terms p we have $\mathsf{nf}(p) \equiv p$.

We illustrate the normal form computation on the branch prediction example. Let $\mathsf{h} = 2$ and consider $(t \wedge^1 f).(l \vee^2 r)$. We use the distributivity of concatenation over choice for terms of lower urgency, and afterwards add the missing unary choices:

$$\begin{aligned} & (t \wedge^1 f).(l \vee^2 r) \\ \overset{\text{(D1)}}{\equiv}\ & (t \wedge^1 f).l \vee^2 (t \wedge^1 f).r \\ \overset{\text{(D2)}}{\equiv}\ & (t.l \wedge^1 f.l) \vee^2 (t.r \wedge^1 f.r) \\ \overset{\text{(L6)}}{\equiv}\ & [\wedge^2 \vee^1 (t.l \wedge^1 f.l)] \vee^2 [\wedge^2 \vee^1 (t.r \wedge^1 f.r)] \ . \end{aligned}$$

By generalizing the example, we get:

Lemma 5: For any term $p \in \mathbf{T}$ without non-terminals, we can find $\mathsf{nf}(p) \in NF_{\mathsf{h}}$ with $p \equiv \mathsf{nf}(p)$.

We show in more detail how to normalize non-terminals.

Lemma 6: For all non-terminals $A \in N$, we can find a $\mathsf{nf}(A) \in NF_{\mathsf{h}}$ with $\mathsf{nf}(A) \equiv A$.

Proof: We proceed by a transfinite Kleene iteration. By induction on α, we construct normal form terms $A^{(\alpha)} \in NF_{\mathsf{h}}$ for all non-terminals A and all ordinals α. We write $p^{(\gamma)}$ to denote $p\{N/N^{(\gamma)}\}$, where $N^{(\gamma)}$ refers to a vector of terms that has $A^{(\gamma)}$ as its A component. We let $A^{(0)} = \mathsf{nf}(\mathsf{err})$ and for all $\alpha > 0$:

$$A^{(\alpha)} = \mathsf{nf}(\bigvee^{\mathsf{h}} \{E(A)^{(\beta)} \mid \beta < \alpha\}) \ .$$

In both cases, we rely on the normalization from Lemma 5.

We claim that $A^{(\beta)} \sqsubseteq A^{(\alpha)} \sqsubseteq A$ for all ordinals $\beta < \alpha$. By definition, the alternatives available to Eve in $A^{(\beta)}$ are contained in the alterantives available to her in $A^{(\alpha)}$. The former precongruence $A^{(\beta)} \sqsubseteq A^{(\alpha)}$ then follows from a simple application of (L1),(L4), and (L5).

With a transfinite induction we show that for all ordinals α and for all non-terminals A we have $A^{(\alpha)} \sqsubseteq A$. For the base case, we already have $A^{(0)} \equiv \mathsf{err} \sqsubseteq A$. For the inductive case, let α be an ordinal so that for all ordinals $\beta < \alpha$ and all non-terminals A we have $A^{(\beta)} \sqsubseteq A$. Using (REP), we see that $E(A)^{(\beta)} \sqsubseteq E(A)$. So we can apply (L1) to get

$$\bigvee^{\mathsf{h}} \{E(A)^{(\beta)} \mid \beta < \alpha\} \ \sqsubseteq\ \bigvee^{\mathsf{h}} E(A) \overset{\text{(L6)}}{\equiv} E(A) \overset{\text{(FP)}}{\equiv} A \ .$$

It remains to show that $A \sqsubseteq A^{(\gamma)}$ for some ordinal γ. The largest $\sqsubseteq$-chain of strictly increasing elements in NF_{h} has size $|ANF_{\mathsf{h}}|$. Then the largest such chain in $N \to NF_{\mathsf{h}}$ has size $|N||ANF_{\mathsf{h}}|$. As a chain forms a well-ordered set, there is an ordinal γ having at least this size. This means we have $A^{(\gamma)} \equiv A^{(\gamma+1)}$ for all non-terminals, and

$$\begin{aligned} E(A)^{(\gamma+1)} &\overset{\text{(L5)}}{\sqsubseteq} E(A)^{(\gamma)} \ \vee^{\mathsf{h}} \bigvee^{\mathsf{h}} \{E(A)^{(\beta)} \mid \beta < \gamma\} \\ &\overset{\text{(L4)}}{\equiv} \bigvee^{\mathsf{h}} \{E(A)^{(\beta)} \mid \beta < \gamma + 1\} \equiv A^{(\gamma+1)} \equiv A^{(\gamma)} \ . \end{aligned}$$

Applying (LFP), we get $A \sqsubseteq A^{(\gamma)}$ for all $A \in N$. ∎

Proposition 2 follows from Lemma 5 and Lemma 6. Let p be the term of interest. With Lemma 6, we compute a normal form term $\mathsf{nf}(A) \equiv A$ for every non-terminal $A \in N$. We use these normal form terms to replace the non-terminals in p, denoted by $p\{N/\mathsf{nf}(N)\}$. This needs repeated applications of (L1) and congruence. Finally, we apply Lemma 5 to normalize the latter term and obtain $\mathsf{nf}(p\{N/\mathsf{nf}(N)\}) \equiv p$.

VI. Completeness

The main result in this section is the completeness of the axiomatic precongruence.

Proposition 3 (Completeness): $p \preceq q$ implies $p \sqsubseteq q$.

The proof relies on two results. The first is the completeness of the specialized axiomatic precongruence.

Proposition 4 (Completeness, Specialized Case): For a right-separating objective $O \subseteq \Sigma^*$, $p \preceq_O q$ implies $p \sqsubseteq_O q$.

The second result is the existence of an objective for which Axiom (S) does not add any relations. Formally, we call an objective O *domain shattering*, if $\sqsubseteq = \sqsubseteq_O$ holds.

Lemma 7 (Existence): There is a domain-shattering and right-separating objective.

Given the two results, the argument for completeness is this. Let $O \subseteq \Sigma^*$ be the domain-shattering and right-separating objective from Lemma 7. We have

$$\preceq \overset{\text{Definition}}{\subseteq} \preceq_O \overset{\text{Prop. 4}}{\subseteq} \sqsubseteq_O \overset{\text{Shattering}}{=} \sqsubseteq \overset{\text{Soundness, Prop. 1}}{\subseteq} \preceq \ .$$

This shows the desired $\sqsubseteq = \preceq$.

A. Completeness Proof, Specialized Case

To show Proposition 4, fix a right-separating $O \subseteq \Sigma^*$. With the normalization in Proposition 2 and soundness of the axiomatic precongruence in Proposition 1, it is sufficient to show completeness for terms in normal form. For $p, q \in NF_{\mathsf{h}}$, we want to show that $p \not\sqsubseteq_O q$ implies $p \not\preceq_O q$ by giving a context that tells them apart. This, however, is difficult as it requires us to understand precisely when the axiomatic congruence fails.

Our way out is to define a less flexible preorder that is easier to handle. The *domination preorder* $\subseteq_O$ only relates normal form terms of the same urgency and owned by the same player. It is defined by induction on the urgency. For $w, v \in NF_0$, we have $w \subseteq_O v$ if $w \leq_O^s v$. For $\mathsf{u} > 0$, we set

$$\bigwedge^{\mathsf{u}} P \subseteq_O \bigwedge^{\mathsf{u}} Q \qquad \text{if } \forall s \in Q.\ \exists r \in P.\ r \subseteq_O s$$
$$\bigvee^{\mathsf{u}} P \subseteq_O \bigvee^{\mathsf{u}} Q \qquad \text{if } \forall r \in P.\ \exists s \in Q.\ r \subseteq_O s\ .$$

Proposition 4 holds with the following lemma.

Lemma 8: Let O be right-separating and $p, q \in NF_{\mathsf{h}}$. Then $p \preceq_O q$ implies $p \subseteq_O q$ and $p \subseteq_O q$ implies $p \sqsubseteq_O q$.

Proof: We focus on the former implication, the latter is simple. The implication trivially holds for terms p that are minimal in the domination preorder. We will need information about the shape of these minimal terms. In urgency zero, minimal are all terms $w \leq_O^s \mathsf{err}$, meaning there is no chance to extend w to a word in O. For higher urgencies $\mathsf{u} > 0$, terms $\bigvee^{\mathsf{u}} P$ are minimal where all elements in P are minimal. Terms $\bigwedge^{\mathsf{u}} P$ are minimal where an element in P is minimal.

To show the implication for terms p and q where p is not minimal, we use *characteristic contexts*. Given a normal form term p, we construct a context $c_p[\bullet]$ so that for all normal form terms q of the same urgency as p and owned by the same player we have:

$$c_p[q] \Downarrow O \quad \text{iff} \quad p \subseteq_O q\ . \tag{1}$$

The implication from $p \preceq_O q$ to $p \subseteq_O q$ indeed follows. Since $p \subseteq_O p$, we obtain $c_p[p] \Downarrow O$ by Equivalence (1). The assumption $p \preceq_O q$ now yields $c_p[q] \Downarrow O$. Hence, again by Equivalence (1), we have $p \subseteq_O q$. For the maximal urgency, we need a special treatment.

It remains to give the construction of $c_p[\bullet]$. It will have the shape $\bullet.t_p$ where t_p is again in normal form. We proceed by induction on the urgency u and need a special case for the maximal urgency h.

Base Case $p \in NF_0$. We define

$$t_p \quad = \quad \bigwedge^{1} \{y \in \Sigma^* \mid p.y \in O\}\ .$$

The set is non-empty as p is not minimal. Moreover, for every normal form term q of urgency zero, Equivalence (1) holds by the definition of $\leq_O^s$.

Inductive Case $p = \bigvee^{\mathsf{u}} P \in NF_{\mathsf{u}}$, $\mathsf{u} < \mathsf{h}$. We define

$$t_p \quad = \quad \bigwedge^{\mathsf{u}+1} \{t_r \mid r \in P \text{ not minimal}\}\ .$$

Note that we can increase the urgency because $\mathsf{u} < \mathsf{h}$. A non-minimal r is guaranteed to exist in P by the assumption that $\bigvee^{\mathsf{u}} P$ itself is not minimal. To prove Equivalence (1), we consider $\bigvee^{\mathsf{u}} Q \in NF_{\mathsf{u}}$ and argue as follows:

$$\begin{aligned}
& (\bigvee^{\mathsf{u}} Q).(\bigwedge^{\mathsf{u}+1} \{t_r \mid r \in P \text{ not minimal}\}) \Downarrow O \\
\text{iff} \quad & \forall r \in P \text{ not minimal}.\ \exists s \in Q.\ s.t_r \Downarrow O \\
\{\text{I.H.}\} \quad \text{iff} \quad & \forall r \in P \text{ not minimal}.\ \exists s \in Q.\ r \subseteq_O s \\
\text{iff} \quad & \forall r \in P.\ \exists s \in Q.\ r \subseteq_O s \\
\text{iff} \quad & \bigvee^{\mathsf{u}} P \ \subseteq_O \ \bigvee^{\mathsf{u}} Q\ .
\end{aligned}$$

The inductive case for $p = \bigwedge^{\mathsf{u}} P \in ANF_{\mathsf{u}}$, $\mathsf{u} \leq \mathsf{h}$, is similar.

Special Case NF_{h}: Since the urgency $\mathsf{h}+1$ is not allowed, we are not able to construct the context in the way we did above. Instead, we show that for $p, q \in NF_{\mathsf{h}}$ with $p \not\subseteq_O q$, there is a term $r \in ANF_{\mathsf{h}}$ where $p.t_r$ is won by Eve and $q.t_r$ is won by Adam. This yields $p \not\preceq q$.

Let $p = \bigvee^{\mathsf{h}} P$ and $q = \bigvee^{\mathsf{h}} Q$ both in NF_{h} with $p \not\subseteq_O q$. By definition of the domination preorder, there is $r \in P$ so that for all $s \in Q$ we have $r \not\subseteq_O s$. The term r cannot be minimal, and hence t_r is guaranteed to exist. We claim that Eve wins O from $p.t_r$ while Adam wins O from $q.t_r$. Note that the leading terms are $\underline{p}.t_r$ resp. $\underline{q}.t_r$. To see that Eve wins from $\underline{p}.t_r$, let her choose $r \in P \subseteq ANF_{\mathsf{h}}$ to reach $r.t_r$. Since $r \subseteq_O r$, Equivalence (1) yields $r.t_r \Downarrow O$. To see that Adam wins from $\underline{q}.t_r$, let Eve choose $s \in Q \subseteq ANF_{\mathsf{h}}$. As $r \not\subseteq_O s$, Equivalence (1) implies that Eve loses from $s.t_r$. ■

B. Construction of Domain-Shattering Objectives

We show Lemma 7, namely that there are right-separating and domain-shattering objectives. For $|\Sigma| > 1$, we take

$$O = \{w.w^{reverse} \mid w \in \Sigma^*\}\ .$$

For $\Sigma = \{a\}$, we let $O = \{a^{(n^2)} \mid n \in \mathbb{N}\}$. It is known that these objectives have a syntactic congruence with singleton classes. Even the syntactic precongruence is the reflexive relation, $\leq_O^s = \cong_O^s$. We give the details in Appendix F, together with the prove that the relations are right-separating.

VII. Decidability and Complexity

We study the decidability and complexity of checking the contextual preorder and its specialized variant. To this end, we leave the setting of infinitary syntax and call p *finitary*, if it refers to a finite set of non-terminals (N, E) and all defining terms $E(A)$ as well as p itself have finite syntax trees. We make the decision problem parametric in the relation R to be checked, and will instantiate R with $\simeq$, $\simeq_O$, and $\preceq_O$:

h-DEC-R
Given: Finitary p, q over Σ, (N, E) of urgency h.
Problem: Does $p\ R\ q$ hold?

The first finding is that already the contextual equivalence is undecidable. The proof is by a reduction from the equivalence problem for context-free languages and the result continues to holds if we fix an alphabet with at least two letters. A consequence is that also the specialized contextual equivalence is undecdable for domain-shattering objectives.

Proposition 5: h-DEC-$\simeq$ and h-DEC-$\simeq_O$ with O domain shattering are undecidable for every h.

Recall that the language $w.w^{reverse}$ is domain shattering, so already context-free objectives lead to undecidability.

Our main result in this section is that for regular objectives the specialized contextual preorder is decidable. We can also give the precise complexity, for which we measure the size of the input in the expected way as $|p| + |q| + |\Sigma| + |E|$. The size of the defining equations is $|E| = \sum_{A \in N} 1 + |E(A)|$. The size of a term is $|\mathsf{skip}| = |\mathsf{err}| = |a| = |A| = 1$ and

$$|p.q| = 1 + |p| + |q| \qquad |\bigcirc^{\mathsf{u}} P| = 1 + \sum_{p \in P} |p|.$$

Theorem 3: Let h be an urgency. For every regular $O \neq \emptyset$, the problem h-DEC-$\preceq_O$ is PTIME-complete.

It is worth noting that the result does not expect the objective to be right-separating. We can indeed decide the specialized contextual preorder $\preceq_O$ for all regular objectives O. Moreover, the lower bound holds no matter which objective is chosen.

It is natural to define a variant h-DEC-$\preceq_*$ of the problem in which also the objective is part of the input and given as a deterministic finite automaton $(\Sigma, Q, i, \delta, \mathcal{F})$. In this case, we use $|O|$ to refer to $|\Sigma| + |Q|$. The following result shows the dramatic influence that the objective has on the complexity.

Theorem 4: h-DEC-$\preceq_*$ is $(2\mathsf{h}-1)-$EXPTIME-complete.

A consequence is that we can also solve the problem of making an observation, denoted here as h-DEC-$\Downarrow$ and assumed to have almost the same input as h-DEC-$\preceq_*$.

Corollary 1: h-DEC-$\Downarrow$ is $(2\mathsf{h}-1)-$EXPTIME-complete.

We use this in Appendix J to derive new decidability results for hyper model checking.

VIII. Upper Bound

We prove the upper bounds claimed in Theorems 3 and 4 through the following statement.

Proposition 6: Given finitary terms p and q and a regular objective O as a DFA, deciding $p \preceq_O q$ can be done in time $(|p| + |q| + |E||N|) \cdot \exp_{2\mathsf{h}-1}(\mathcal{O}(|O|^2))$.

The undecidability result in Proposition 5 shows that the normal form for the axiomatic congruence from Section V is insufficient as a basis for algorithms. The problem is that the normal form terms are typically infinite, and therefore difficult to handle computationally. The source of infinity can be found in the base case: already $NF_0 = \Sigma^* \cup \{\mathsf{err}\}$ is infinite, and this propagates upwards. We realize that the O-specialized axiomatic congruence admits a more refined normal form that is guaranteed to yield finite terms (and finitely many of them). The key idea is to factorize NF_0 using Axiom (S).

We define the set of *O-specialized normal form terms* by induction on the urgency. The base case $SNF_0^O = \Sigma^*_{\mathsf{err}}/_{\cong^s_O}$ are classes of words in the syntactic congruence $\cong^s_O = \leq^s_O \cap \geq^s_O$ induced by O. For $\mathsf{u} > 0$, the definition is almost as before:

$$SANF^O_{\mathsf{u}} = \{\bigwedge^{\mathsf{u}} P \mid \emptyset \neq P \subseteq SNF^O_{\mathsf{u}-1}\}$$
$$SNF^O_{\mathsf{u}} = \{\bigvee^{\mathsf{u}} P \mid \emptyset \neq P \subseteq SANF^O_{\mathsf{u}}\}.$$

Note that since O is regular, the set SNF^O_0 and so all $SANF^O_{\mathsf{u}}$ and SNF^O_{u} are guaranteed to be finite [12]. This is precisely where regularity comes in. Another aspect is that we change the alphabet to having $\cong^s_O$-congruence classes as letters. This can be fixed by working with a representative system, meaning we represent every $\cong^s_O$-class by one of its elements.

We adapt the normalization process from Section V to compute a term in the O-specialized normal form. Only the base case changes, for the inductive cases we merely study the complexity. Interestingly, the overall normalization takes time $2\mathsf{h}$-fold exponential only in the size of the objective, using the common definition $\exp_0(x) = x$ and $\exp_{\mathsf{u}+1}(x) = 2^{\exp_{\mathsf{u}}(x)}$.

Lemma 9: Given a finitary term p and a regular objective $O \subseteq \Sigma^*$ as a DFA, we can compute $\mathsf{nf}_O(p) \in SNF^O_{\mathsf{h}}$ with $\mathsf{nf}_O(p) \equiv_O p$ in time $(|p| + |E||N|) \cdot \exp_{2\mathsf{h}-1}(\mathcal{O}(|SNF^O_0|))$. We have $|SNF^O_{\mathsf{h}}| = \exp_{2\mathsf{h}}(\mathcal{O}(|SNF^O_0|))$.

The result already allows us to decide the O-specialized contextual preorder as follows. Since we cannot assume the objective to be right-separating, the algorithm cannot rely on a full abstraction result. Instead, we have to evaluate $p \preceq_O q$ directly, by iterating over contexts. What makes this possible is the combination of our proof technique for soundness in Lemma 2 and the O-specialized normal form just introduced. With Lemma 2, we do not have to iterate over all contexts to show $p \preceq_O q$, but only over contexts of the form $r.\bullet.s$. With Lemma 9, the terms r and s can be normalized.

Corollary 2: Consider a regular O and finitary p, q. Then $p \preceq_O q$ if and only if for all $c[\bullet] = r.\bullet.s$ with $r \in SNF^O_{\mathsf{h}-1}$, $s \in SNF^O_{\mathsf{h}}$ we have $\mathsf{nf}_O(c[p]) \Downarrow O$ implies $\mathsf{nf}_O(c[q]) \Downarrow O$.

Unfortunately, the algorithm formulated in the corollary is slower than the promised upper bound by two exponents because SNF^O_{h} contains $\exp_{2\mathsf{h}+1}(\mathcal{O}(|O|^2))$ many terms. To overcome the problem, the first step is to reduce the number of contexts that have to be considered. The idea is to factorize the contexts along their solution spaces. The solution space of a context is the set of terms p for which $c[p] \Downarrow O$ holds. When checking the specialized preorder $\preceq_O$, the job of a context $c[\bullet]$ is to disprove $p \preceq_O q$ by showing $c[p] \Downarrow O$ and $c[q] \not\Downarrow O$. Hence, when two contexts have the same solution space, it suffices to consider one of them.

What makes the solution space equivalence algorithmically interesting is that (i) it is coarse enough to save an exponent and (ii) we can directly compute with equivalence classes of contexts. The key insight behind both statements is that the solution space of a context can be represented in a convenient way: it is the $\sqsubseteq_O$-upward closure of a so-called *characteristic term*. This is a novel concept that deserves a definition.

Definition 5: Term p is *characteristic for* $c[\bullet]$ *wrt.* $O \subseteq \Sigma^*$, if for all $q \in \mathbf{T}$ we have $c[q] \Downarrow O$ if and only if $p \sqsubseteq_O q$.

We will show that there are only $\exp_{2\mathsf{h}}(\mathcal{O}(|O|^2))$ many characteristic terms for contexts of the form $r.\bullet.s$. Moreover, we can compute the characteristic terms directly, without building up the corresponding contexts. This saves an exponent in the complexity: we modify the algorithm in Corollary 2 to iterate through characteristic terms rather than contexts.

We save another exponent in the runtime of our algorithm by a more compact representation of the terms in O-specialized normal form. One exponent in the size of SNF^O_{h} is inherited from the base case, where already $|SNF^O_0| = \exp_1(\mathcal{O}(|O|^2))$. We use the fact that each class of the syntactic congruence can be represented by a function $Q \to Q$ between the states of the objective DFA [12]. The key idea is to see these functions as sets of pairs and decompose them into their elements. We simulate a function, say for letter a, by letting Eve choose a state change (p, q) with $\delta(p, a) = q$. We thus represent the congruence class $[a] \in \Sigma^*_{\mathsf{err}}/\cong^s_O$ by an angelic choice of urgency 1 over letters from the alphabet $Q \times Q$. Since the objective is given by a deterministic automaton, Adam does not gain any knowledge by seeing Eve's choice. The low urgency, in turn, makes sure Eve can see the full run of the DFA up to the current position when making her choice, and she will lose if she picks inconsecutive transitions. The modification propagates to the normal form terms, and requires a (simple) modification of the objective. The new objective $\mathsf{tr}(O)$ has a syntactic congruence with an exponentially smaller index, namely $|SNF^{\mathsf{tr}(O)}_0| = \mathcal{O}(|O|^2)$.

To argue about the complexity, we start with the compact term representation before introducing characteristic terms.

A. Compact Term Representation

Recall that the idea behind our compact term representation is to translate syntactic congruence classes $[a] : Q \to Q$ into angelic choices over state changes. With a representative system for the syntactic congruence, we can assume the source alphabet is Σ rather than $\Sigma^*_{\mathsf{err}}/\cong^s_O$. The translation of term p over Σ is the term $\mathsf{tr}(p)$ over the alphabet Q^2 defined by

$$\begin{aligned} \mathsf{tr}(a) &= \bigvee\nolimits^{1}\{(p,q) \mid \delta(p,a) = q\} & \mathsf{tr}(\mathsf{err}) &= \mathsf{err} \\ \mathsf{tr}(\bigcirc^{\mathsf{u}} P) &= \bigcirc^{\mathsf{u}}\{\mathsf{tr}(p) \mid p \in P\} & \mathsf{tr}(\mathsf{skip}) &= \mathsf{skip} \\ \mathsf{tr}(p.q) &= \mathsf{tr}(p).\mathsf{tr}(q) & \mathsf{tr}(A) &= A\,, \end{aligned}$$

and we also translate the equations, $E_{\mathsf{tr}}(A) = \mathsf{tr}(E(A))$. The new alphabet calls for a translation of the objective. The DFA $\mathsf{tr}(O)$ is obtained from O by modifying the transitions and adding a failure state $\bot$. We let $\mathsf{tr}(\delta)(p,(q,r)) = r$ if $p = q$ and $\mathsf{tr}(\delta)(p,(q,r)) = \bot$ if $p \neq q$. The accepting and initial states remain the same. The translation is faithful when it comes to our notion of observable behavior.

Lemma 10: $p \Downarrow O$ if and only if $\mathsf{tr}(p) \Downarrow \mathsf{tr}(O)$.

By combining Lemma 10 and Corollary 2, we can decide the specialized contextual preorder $p \preceq_O q$ by checking whether $c[\mathsf{tr}(p)] \Downarrow \mathsf{tr}(O)$ implies $c[\mathsf{tr}(q)] \Downarrow \mathsf{tr}(O)$ for all contexts $\mathsf{tr}(s).\bullet.\mathsf{tr}(t)$ with $s \in SNF^O_{\mathsf{h}-1}$ and $t \in SNF^O_{\mathsf{h}}$. The problem with this algorithm is that, for complexity reasons, we cannot work with an explicit translation of terms in specialized normal form. To overcome the problem, an attempt would be to generalize the above set of contexts to all $s.\bullet.t$ with $s \in SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}$ and $t \in SNF^{\mathsf{tr}(O)}_{\mathsf{h}}$. However, the set $SNF^{\mathsf{tr}(O)}_{\mathsf{h}}$ contains terms over the new alphabet Q^2 that do not result from a translation of an SNF^O_{h} term. Unfortunately, these extra contexts may, incorrectly so, disprove the specialized contextual preorder of interest.

Our solution is to come up with a direct construction for the image of SNF^O_{h} under $\mathsf{nf}_{\mathsf{tr}(O)} \circ \mathsf{tr}$, the translation followed by a normalization. This is the appropriate subset of $SNF^{\mathsf{tr}(O)}_{\mathsf{h}}$ over which we should form contexts. The idea behind the construction is to explicitly translate the urgency 1 terms in specialized normal form, and build up the higher orders in the standard way. We define the *translated O-specialized normal form* terms (with $\mathsf{u} > 1$) by

$$\begin{aligned} TNF^{\mathsf{tr}(O)}_0 &= SNF^{\mathsf{tr}(O)}_0 \qquad TNF^{\mathsf{tr}(O)}_1 = \mathsf{nf}_{\mathsf{tr}(O)}(\mathsf{tr}(SNF^O_1)) \\ TANF^{\mathsf{tr}(O)}_{\mathsf{u}} &= \{\bigwedge\nolimits^{\mathsf{u}} P \mid \emptyset \neq P \subseteq TNF^{\mathsf{tr}(O)}_{\mathsf{u}-1}\} \\ TNF^{\mathsf{tr}(O)}_{\mathsf{u}} &= \{\bigvee\nolimits^{\mathsf{u}} P \mid \emptyset \neq P \subseteq TANF^{\mathsf{tr}(O)}_{\mathsf{u}}\}\,. \end{aligned}$$

The set of contexts we should iterate over is thus

$$\mathbf{C}_{\mathsf{tr}(O)} \;=\; \{r.\bullet.s \mid r \in TNF^{\mathsf{tr}(O)}_{\mathsf{h}-1} \text{ and } s \in TNF^{\mathsf{tr}(O)}_{\mathsf{h}}\}\,.$$

The argumentation leads to the following algorithm for checking the specialized contextual preorder.

Proposition 7: Consider a regular objective O and finitary terms p, q. Then $p \preceq_O q$ if and only if for all $c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}$, we have that $c[\mathsf{tr}(p)] \Downarrow \mathsf{tr}(O)$ implies $c[\mathsf{tr}(q)] \Downarrow \mathsf{tr}(O)$.

The benefit of Proposition 7 over Corollary 2 is that the translated normal form terms are in $\mathsf{tr}(O)$-specialized normal form, $TNF^{\mathsf{tr}(O)}_{\mathsf{h}} \subseteq SNF^{\mathsf{tr}(O)}_{\mathsf{h}}$, so we inherit the size bound.

Lemma 11: $|TNF^{\mathsf{tr}(O)}_{\mathsf{h}}| \leq \exp_{2\mathsf{h}}(\mathcal{O}(|Q|^2))$.

B. Characteristic Terms

With Proposition 7, we need to iterate over 2h-exponentially many contexts. We now eliminate another exponent by factorizing the contexts with the help of characteristic terms. Recall that term p is characteristic for context $c[\bullet]$ wrt. $\mathsf{tr}(O)$, if its $\sqsubseteq_{\mathsf{tr}(O)}$-upward closure is the solution space of the context: for all q we have $p \sqsubseteq_{\mathsf{tr}(O)} q$ if and only if $c[q] \Downarrow \mathsf{tr}(O)$.

For the contexts $c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}$ we just defined, giving Adam a choice over the solution space yields a characteristic term:

$$\chi(c[\bullet]) \;=\; \bigwedge\nolimits^{\mathsf{h}}\{p \in SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1} \mid c[p] \Downarrow \mathsf{tr}(O)\}\,. \qquad (2)$$

To see that the term is characteristic indeed, we rely on the domination preorder introduced in the completeness proof.

Lemma 12: Term $\chi(c[\bullet])$ is characteristic for $c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}$ wrt. $\mathsf{tr}(O)$ and can be computed in time $\exp_{2\mathsf{h}-1}(\mathcal{O}(|Q|^2))$.

Proof: To prove that $\chi(c[\bullet])$ is characteristic, consider term $p = \bigvee^{\mathsf{h}}_{i \in I} \bigwedge^{\mathsf{h}} P_i$ in $\mathsf{tr}(O)$-specialized normal form. As p is immediate for $c[\bullet]$, we get $c[p] \Downarrow \mathsf{tr}(O)$ if and only if there is an index $i \in I$ so that for all $q \in P_i$ we have $c[q] \Downarrow \mathsf{tr}(O)$. This can be shown to be equivalent to the domination preorder

$\bigvee^{\mathsf{h}} \bigwedge^{\mathsf{h}} \{p \in SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1} \mid c[p] \Downarrow \mathsf{tr}(O)\} \subseteq_{\mathsf{tr}(O)} \bigvee^{\mathsf{h}}_{i\in I} \bigwedge^{\mathsf{h}} P_i$. For the terms at hand, this domination preorder is equivalent to $\bigwedge^{\mathsf{h}} \{p \in SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1} \mid c[p] \Downarrow \mathsf{tr}(O)\} \sqsubseteq_{\mathsf{tr}(O)} \bigvee^{\mathsf{h}}_{i\in I} \bigwedge^{\mathsf{h}} P_i$, even if the objective is not right-separating.

To compute the characteristic term, we have to check, for every term $p \in SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}$, whether $c[p] \Downarrow \mathsf{tr}(O)$ holds. Such a check requires a normalization of $c[p]$, followed by a polynomial-time evaluation of the resulting term. The normalization takes time $|c[p]| \cdot \exp_{2\mathsf{h}-1}(\mathcal{O}(|Q|^2))$, Lemma 9. The dominating factor in $|c[p]|$ is the size of the $TNF^{\mathsf{tr}(O)}_{\mathsf{h}}$ term in the context, which is bounded by $\exp_{2\mathsf{h}-1}(\mathcal{O}(|Q|^2))$. There are $\exp_{2\mathsf{h}-2}(\mathcal{O}(|Q|^2))$ terms p we have to go through. The overall runtime is thus bounded by $\exp_{2\mathsf{h}-1}(\mathcal{O}(|Q|^2))$. ■

Let $\chi(\mathbf{C}_{\mathsf{tr}(O)}) = \{\chi(c[\bullet]) \mid c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}\}$ denote the set of characteristic terms. As these terms belong to $SANF^{\mathsf{tr}(O)}_{\mathsf{h}}$, we inherit the following bound.

Lemma 13: $|\chi(\mathbf{C}_{\mathsf{tr}(O)})| \leq \exp_{2\mathsf{h}-1}(\mathcal{O}(|O|^2))$.

Compared to Lemma 11, there are exponentially fewer characteristic terms than contexts. To decide $p \preceq_O q$, we thus intend to iterate over all $x \in \chi(\mathbf{C}_{\mathsf{tr}(O)})$ and check whether $x \preceq_{\mathsf{tr}(O)} p$ implies $x \preceq_{\mathsf{tr}(O)} q$. We will use the domination preorder for these checks.

Lemma 14: Given terms $p, q \in SNF^{\mathsf{tr}(O)}_{\mathsf{h}}$, we can decide $p \subseteq_{\mathsf{tr}(O)} q$ in time $|p| \cdot |q|$.

There is a last obstacle: we do not know the characteristic terms, like we did not know the translated normal form terms above. Going through all contexts and determining the characteristic terms is prohibitively expensive. Generalizing from $\chi(\mathbf{C}_{\mathsf{tr}(O)})$ to $SANF^{\mathsf{tr}(O)}_{\mathsf{h}}$ is incorrect. The way out is to give a direct construction of the characteristic terms.

The key insight behind the direct construction is that the characteristic terms satisfy the following equation, where we have $s \in SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}$, $T \subseteq SANF^{\mathsf{tr}(O)}_{\mathsf{h}}$, and $c[\bullet] = s.\bullet.\bigvee^{\mathsf{h}} T$:

$$\chi(c[\bullet]) \;\equiv_{\mathsf{tr}(O)}\; \bigwedge^{\mathsf{h}} \{\chi(s.\bullet.t) \mid t \in T\}\ . \tag{3}$$

The equation follows from Equation (2), Appendix I. The impact of Equation (3) may not be immediate: we still have to make sure to construct the characteristic term for every set $T \subseteq SANF^{\mathsf{tr}(O)}_{\mathsf{h}}$. What the equation does is to give us an inductive formulation of the characteristic terms which allows us to compute the set of all characteristic terms in a fixed point. We first construct the characteristic terms for singleton sets $|T| = 1$. Then we conjoin the characteristic terms as prescribed by Equation (3) to obtain the characteristic terms for sets of size $|T| \leq 2$. We repeat the latter conjunction until we reach a fixed point. Throughout the process, we work up to $\equiv_{\mathsf{tr}(O)}$. With Lemma 13, the sets we compute with have size at most $\exp_{2\mathsf{h}-1}(\mathcal{O}(|O|^2))$. Moreover, we are guaranteed to reach the fixed point after at most $\exp_{2\mathsf{h}-1}(\mathcal{O}(|O|^2))$ steps. To state the correctness, define for $P \subseteq SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}$ and $Q \subseteq SANF^{\mathsf{tr}(O)}_{\mathsf{h}}$:

$$\mathbb{X}(P,Q) = \bigcup_{s\in P, T\subseteq Q} \{\chi(s.\bullet.\bigvee^{\mathsf{h}} T)\}\ .$$

Lemma 15: $\chi(\mathbf{C}_{\mathsf{tr}(O)}) = \mathbb{X}(TNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}, TANF^{\mathsf{tr}(O)}_{\mathsf{h}})$. The set can be computed in time $\exp_{2\mathsf{h}-1}(\mathcal{O}(|O|^2))$.

The following proposition yields the overall algorithm.

Proposition 8: Consider a regular O and finitary p, q. Then $p \preceq_O q$ if and only if for all $x \in \mathbb{X}(TNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}, TANF^{\mathsf{tr}(O)}_{\mathsf{h}})$, we have that $x \subseteq_{\mathsf{tr}(O)} \mathsf{nf}_{\mathsf{tr}(O)}(p)$ implies $x \subseteq_{\mathsf{tr}(O)} \mathsf{nf}_{\mathsf{tr}(O)}(q)$. The domination preorder is sound for checking the axiomatic preorder even for objectives that fail to be right-separating because we have characteristic terms on the left. The time for computing the characteristic terms is given in Lemma 15. The normalization is Lemma 9, and we make use of the fact that the syntactic congruence of $\mathsf{tr}(O)$ has size quadratic in $|Q|$. By Lemma 14, the comparison takes quadratic time. This concludes the proof of Proposition 6.

IX. Lower Bound

We prove the lower bound given in Theorem 4 with a reduction from context-bounded multi-pushdown games, a concurrent programming model the complexity of which is well-understood [14], [15]. The proof of the lower bound given in Theorem 3 can be found in Appendix H.

A. Multi-Pushdown Games

We introduce multi-pushdown games trimmed to our needs. A b-*context-bounded 2-stack pushdown game (*b-*2PDG)* is a tuple $(Q, E, p_0, F, \Gamma, \delta)$ consisting of a finite set of states Q, a set of states $E \subseteq Q$ owned by Eve, an initial state p_0, a set of goal states $F \subseteq Q$, a stack alphabet Γ, and a set of transitions $\delta \subseteq Q \times Op \times Q$. Transitions are annotated by a stack operation from $Op = \Gamma \times \Gamma^{\leq 2} \cup \{\mathrm{nx}\}$. With (a, w), we pop a from and push w to the active stack. With nx, we change the active stack, called a context switch. We assume there is a bottom of stack symbol $\$ \in \Gamma$ that is never removed.

The semantics of a b-2PDG is a game arena $(CF, \rightarrow, \mathsf{own})$ with a reachability objective CF_F for Eve. The positions are configuration from $CF = Q \times [0, \mathsf{b}] \times \Gamma^* \times \Gamma^*$. A configuration $(p, \mathsf{k}, s_1, s_0)$ stores the current state p, the number of context switches k that have occurred so far, and the contents of the two stacks. Stack s_0 is active after an even number of context switches, stack s_1 is active when k is odd. The owner and moves are defined as expected, there are no context switches beyond b, and we assume there are no deadlocks. The objective is $CF_F = F \times \{\mathsf{b}\} \times \Gamma^* \times \Gamma^*$, meaning we reach a goal state and have exhausted the context switches. Plays, strategies, and winning are defined like for urgency programs. The task is to decide whether Eve has a strategy to win from $(p_0, 0, \$, \$)$.

Theorem 5: [15] b-2PDG are $(\mathsf{b}-2)-$EXPTIME-complete.

B. Reduction

The reduction is in two steps, we first reduce 2PDG to the problem of making an observation:

Proposition 9: Given a $(2\mathsf{h}+1)$-2PDG PD, we can compute in poly time p over Σ and (N, E) of maximal urgency h and an objective DFA O so that Eve wins PD if and only if $p \Downarrow O$.

We now reduce the problem of making an observation to the specialized contextual equivalence. Indeed, $p \Downarrow O$ is the same

as to check $\chi_O(\bullet) \preceq_O p$, where $\chi_O(\bullet)$ is the characteristic term of the empty context formed for objective O. The problems is that the characteristic term may be exponential. We utilize the trick from Section VIII-A.

Lemma 16: Given p over Σ and (N, E), and objective O, we can compute in poly time $\chi_{\mathsf{tr}(O)}(\bullet)$, $\mathsf{tr}(p)$, and $\mathsf{tr}(O)$ so that $p \Downarrow O$ if and only if $\chi(\bullet) \preceq_{\mathsf{tr}(O)} \mathsf{tr}(p)$.

We sketch the proof of Proposition 9. We encode positions $(p, \mathsf{k}, x \dots \$, y \dots \$)$ of the 2PDG as urgency terms

$$\mathrm{H}^{\mathsf{w}}.\underbrace{g.x^{\mathsf{u}} \dots \$^{\mathsf{u}}.@}_{s_1}.\underbrace{h.y^{\mathsf{v}} \dots \$^{\mathsf{v}}.@}_{s_0}.$$

Stack symbols $x \in \Gamma$ are represented by terms x^{u} of urgency u. Terminal @ marks the end of a stack content encoding. The terms g and h represent the history of the play. Finally, H^{w} implements context switches. The construction controls w, u, and v so that the top of the active stack is leading.

The top of the active stack allows the game to proceed as

$$\dots g.(\underbrace{\bigvee_{p \in Q} \bigcirc_{t \in \delta_{p,x}} \langle t \rangle^{\mathsf{u}}}_{\text{Rewritten from } x^{\mathsf{u}}}) \dots \$^{\mathsf{u}}.@ \dots \to \dots g.\langle t \rangle^{\mathsf{u}}_x \dots \$^{\mathsf{u}}.@ \dots$$

Eve selects the current state $p \in Q$. Then the player owning this state selects the next transition. We use $\delta_{p,x} \subseteq \delta$ to denote the set of transitions from state p with top of stack symbol x. The set is non-empty because the 2PDG does not deadlock. The term $\langle t \rangle^{\mathsf{u}}$ of the chosen transition contains terminals which join history g to record the state change and the urgency u. The objective O is a product DFA that reads the terminals for each urgency separately and enforces consistency with the 2PDA transitions.

Push/pop operations modify the active stack encoding in the expected way. For context switches, the leading term must swap the stack. To implement this, we use a decrement processs on the now no longer active stack. We define stack symbols x^{u} as $\langle \to \mathrm{st} \rangle^{\mathsf{u}}.r \vee^{\mathsf{u}} \langle \to \mathrm{nx} \rangle^{\mathsf{u}}.x^{\mathsf{u}-1}$, where r is the choice of the next transition explained above. The decrement process relies on the alternative $\langle \to \mathrm{nx} \rangle^{\mathsf{u}}.x^{\mathsf{u}-1}$, which replaces x^{u} by $x^{\mathsf{u}-1}$. A snapshot of the decrement process is

$$\dots g'.\langle \mathrm{nx} \rangle \underbrace{\overbrace{\dots \langle \to \mathrm{nx} \rangle.x^{\mathsf{u}-1}_{i-1}}^{\text{Progression of the leading term}}}_{\text{Urgency } \mathsf{u}-1} .\underline{x^{\mathsf{u}}_i} \dots \$^{\mathsf{u}}@ \dots$$

The terminals $\langle \to \mathrm{st} \rangle$, $\langle \to \mathrm{nx} \rangle$, $\langle \mathrm{nx} \rangle$, and @ allow the objective to check the decrement process for correctness.

Each urgency simulates two contexts. Since we do not need to access the odd stack before the first context switch, we only generate this stack when it is first accessed. This allows us to simulate three contexts with the maximal urgency. In total, the construction simulates $2\mathsf{h} + 1$ contexts with urgency h.

X. Related Work

We motivated urgency annotations through hyperproperties. Hyper model checking remains decidable if one language is context-free and the others are regular [5]. We can model this fragment with urgency programs. Gutsfeld et al. [16] propose asynchronous hyperproperties and prove positive results for model checking under bounds on asynchrony and context switching. We believe we can capture this fragment as well, but leave a thorough study to future work. Urgency programs can also serve as a recursive game model with imperfect information. This is interesting because the canonical pushdown games with imperfect information [17] are known to be undecidable even under strong restrictions [18]. Compared to game models with perfect information, our normal form in Section V resembles Walukiewicz's procedure summaries [19], though urgency calls for new techniques.

The goal of effective denotational semantics [20], [21] is to derive from a specification O a denotational semantics $\mathcal{D}_O$ so that $\mathcal{D}_O(p)$ answers the question of whether program p satisfies specification O. Salvati and Walukiewicz studied the expressiveness of extremal fixed point semantics [22], [23] and succeeded in giving an effective denotational semantics that captures the higher-order model-checking problem [24]. Grellois and Melliès developed a link between the intersection type system for higher-order model checking [25] and models of linear logic [26], which they generalized [27], [28] to solve logical reflection [29] and selection problems [30], [31]. Satisfying a specification means $p \Downarrow O$, so there is a close link to our axiomatization of the specialized contextual preorder: in Appendix M, we show how our axiomatization induces a denotational semantics that is effective for finitary programs. A comparable construction does not appear in the above works. It is interesting because it guides the study to the interplay between the operators. Urgencies are also new.

The study of contextual equivalence [6] has lead to the development of game semantics [32], [33]. Game semantics make explicit the interaction of a program with its environment and, in their operational formulation, take the form of a language [7]. Our approach to full abstraction only refers to the program term, though there is an idea of iteration behind axiom (LFP). The language-theoretic understanding of game semantics has been the basis for deciding contextual equivalence of third-order programs [34]. We decide the specialized contextual preorder by explicitly evaluating the observable in a carefully chosen set of contexts.

Urgency annotations are related to prorities in process algebra [35]. The key difference is that priorities preserve the program order while urgencies do not. Out-of-program-order execution is needed for modeling hyperproperties. At the same time, it brings a new form of unbounded memory to the semantics (the unresolved choices) that our axiomatization explains how to handle. Also alternation [1] (angelic and demonic choice) is not common in process algebra. Indeed, we have not found a study of alternation from the perspective of contextual equivalence. It is the special case of our work when the urgency is $\mathsf{h} = 1$.

Our context lemma for proving soundness has relatives in process algebra and semantics [36]. Characteristic objects (here, contexts and terms) have appeared as early as [37]. Our contribution is to adapt the general idea to our setting.

XI. Conclusion and Future Work

We presented urgency annotations for alternating choices as a new programming construct. Urgency annotations allow the choices to execute out-of-program-order and were designed to capture the hyperproperties recently popular in security. Our contribution was a study of the standard notions of contextual equivalence for urgency programs. We gave sound and complete axiomatizations and showed that the specialized contextual preorder (for a regular observable) is $(2h-1)$-EXPTIME-complete if the observable is part of the input resp. PTIME-complete if it is fixed. These result can be used to obtain new computability results for game models and program synthesis tasks with imperfect information and recursion.

Urgency programs are defined over finite words, and the next step is to extend them to infinite words. The challenge is to get the semantics of right. If we define the semantics via infinite unrollings, then it is unclear how to ever switch from choices with higher to choices with lower urgency. Instead, it seems appropriate to work with program terms that contain an ω-operator and can be rewritten a finite number of times. This, however, calls for a different set of algebraic techniques [38].

References

[1] A. K. Chandra and L. J. Stockmeyer, "Alternation," in *FOCS*. IEEE, 1976, pp. 98–108.

[2] M. R. Clarkson and F. B. Schneider, "Hyperproperties," *JCS*, vol. 18, no. 6, pp. 1157–1210, 2010.

[3] G. Barthe, P. R. D'Argenio, and T. Rezk, "Secure information flow by self-composition," in *CSFW*. IEEE, 2004, pp. 100–114.

[4] B. Finkbeiner, M. N. Rabe, and C. Sánchez, "Algorithms for model checking HyperLTL and HyperCTL*," in *CAV*, ser. LNCS, vol. 9206. Springer, 2015, pp. 30–48.

[5] A. Pommellet and T. Touili, "Model-checking hyperltl for pushdown systems," in *SPIN*, ser. LNCS, vol. 10869. Springer, 2018, pp. 133–152.

[6] R. Milner, "Fully abstract models of typed *lambda*-calculi," *TCS*, vol. 4, no. 1, pp. 1–22, 1977.

[7] G. Jaber and A. S. Murawski, "Complete trace models of state and control," in *ESOP*, ser. LNCS, vol. 12648. Springer, 2021, pp. 348–374.

[8] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks: Exploiting speculative execution," in *SP*. IEEE, 2019, pp. 1–19.

[9] D. A. Martin, "Borel determinacy," *AMATH*, vol. 102, no. 2, pp. 363–371, 1975.

[10] M. Hennessy and R. Milner, "Algebraic laws for nondeterminism and concurrency," *JACM*, vol. 32, no. 1, pp. 137–161, 1985.

[11] J. A. Bergstra, A. Ponse, and S. A. Smolka, Eds., *Handbook of Process Algebra*. Elsevier, 2001.

[12] M. O. Rabin and D. S. Scott, "Finite automata and their decision problems," *IBM J. Res. Dev.*, vol. 3, no. 2, pp. 114–125, 1959.

[13] G. Birkhoff, *Lattice Theory*. AMS, 1967.

[14] S. Qadeer and J. Rehof, "Context-bounded model checking of concurrent software," in *TACAS*, ser. LNCS, vol. 3440. Springer, 2005, pp. 93–107.

[15] R. Meyer and S. van der Wall, "On the complexity of multi-pushdown games," in *FSTTCS*, ser. LIPIcs, vol. 182. Dagstuhl, 2020, pp. 52:1–52:35.

[16] J. O. Gutsfeld, M. Müller-Olm, and C. Ohrem, "Automata and fixpoints for asynchronous hyperproperties," *PACMPL*, vol. 5, no. POPL, pp. 1–29, 2021.

[17] B. Aminof, A. Legay, A. Murano, O. Serre, and M. Y. Vardi, "Pushdown module checking with imperfect information," *IC*, vol. 223, pp. 1–17, 2013.

[18] L. Bozzelli, "New results on pushdown module checking with imperfect information," in *GandALF*, ser. EPTCS, vol. 54, 2011, pp. 162–177.

[19] I. Walukiewicz, "Pushdown processes: Games and model-checking," *IC*, vol. 164, no. 2, pp. 234–263, 2001.

[20] K. Aehlig, "A finite semantics of simply-typed lambda terms for infinite runs of automata," *LMCS*, vol. 3, no. 3, 2007.

[21] K. Terui, "Semantic evaluation, intersection types and complexity of simply typed lambda calculus," in *RTA*, ser. LIPIcs, vol. 15. Dagstuhl, 2012, pp. 323–338.

[22] S. Salvati and I. Walukiewicz, "Using models to model-check recursive schemes," *LMCS*, vol. 11, no. 2, 2015.

[23] I. Walukiewicz, "Lambda Y-calculus with priorities," in *LICS*. IEEE, 2019, pp. 1–13.

[24] S. Salvati and I. Walukiewicz, "A model for behavioural properties of higher-order programs," in *CSL*, ser. LIPIcs, vol. 41. Dagstuhl, 2015, pp. 229–243.

[25] N. Kobayashi and C. L. Ong, "A type system equivalent to the modal mu-calculus model checking of higher-order recursion schemes," in *LICS*. IEEE, 2009, pp. 179–188.

[26] C. Grellois and P. Melliès, "Indexed linear logic and higher-order model checking," in *ITRS*, ser. EPTCS, vol. 177, 2014, pp. 43–52.

[27] ——, "An infinitary model of linear logic," in *FoSSaCS*, ser. LNCS, vol. 9034. Springer, 2015, pp. 41–55.

[28] C. Grellois, "Semantics of linear logic and higher-order model-checking," Ph.D. dissertation, Université Paris Diderot, 2016.

[29] C. H. Broadbent, A. Carayol, C. L. Ong, and O. Serre, "Recursion schemes and logical reflection," in *LICS*. IEEE, 2010, pp. 120–129.

[30] A. Carayol and O. Serre, "Collapsible pushdown automata and labeled recursion schemes: Equivalence, safety and effective selection," in *LICS*. IEEE, 2012, pp. 165–174.

[31] A. Haddad, "Model checking and functional program transformations," in *FSTTCS*, ser. LIPIcs, vol. 24. Dagstuhl, 2013, pp. 115–126.

[32] S. Abramsky, R. Jagadeesan, and P. Malacaria, "Full abstraction for PCF," *IC*, vol. 163, no. 2, pp. 409–470, 2000.

[33] J. M. E. Hyland and C. L. Ong, "On full abstraction for PCF: I, II, and III," *IC*, vol. 163, no. 2, pp. 285–408, 2000.

[34] A. S. Murawski and I. Walukiewicz, "Third-order idealized ALGOL with iteration is decidable," in *FoSSaCS*, ser. LNCS, vol. 3441. Springer, 2005, pp. 202–218.

[35] R. Cleaveland, G. Lüttgen, and V. Natarajan, "Priority in process algebra," in *Handbook of Process Algebra*. Elsevier, 2001, pp. 711–765.

[36] A. Pitts and I. Stark, "Operational reasoning for functions with local state," in *Higher Order Operational Techniques in Semantics*. CUP, 1998, pp. 227–273.

[37] R. McNaughton and S. A. Papert, *Counter-Free Automata*. MIT Press, 1971.

[38] T. Wilke, "An algebraic theory for regular languages of finite and infinite words," *Algebra and Computation*, vol. 3, no. 4, pp. 447–489, 1993.

[39] L. M. Goldschlager, "The monotone and planar circuit value problems are log space complete for P," *ACM SIGACT News*, vol. 9, no. 2, pp. 25–29, 1977.

[40] O. M. Committee. Openssl. [Online]. Available: openssl.org

[41] P. C. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems," in *CRYPTO*, ser. LNCS, vol. 1109. Springer, 1996, pp. 104–113.

[42] D. Brumley and D. Boneh, "Remote timing attacks are practical," in *USENIX Security*. USENIX Association, 2003.

APPENDIX

A. Handling Infinitary Syntax

For the development of the section, we call a term an h-term when no choice subterm has urgency higher than h and the expansion of each non-terminal is an h-term. Infinitary syntax requires care to make sure that set theoretic concepts remain sound. The problem lies on the unbounded nature of the choice operator. If we were to allow choices to range over *arbitrary sets* of terms, the class of all terms would no longer form a set. To see this, suppose that the class of terms with unbounded choice were a set $\mathbf{T}$. Then, we could build the term $\bigvee^{\mathsf{u}} \mathbf{T} \in \mathbf{T}$, which contradicts the Axiom of Regularity. But for our development in the paper, it is a convenience to have $\mathbf{T}$ as a set and not be distracted by subtle differences of classes and sets. For this reason, we impose a restriction on the terms, that does not hinder our developments. It is defined over the structural depth of a term, which is

$$\mathsf{depth}(A) = \mathsf{depth}(a) = \mathsf{depth}(\mathsf{err}) = \mathsf{depth}(\mathsf{skip}) = 0$$

$$\mathsf{depth}(p.q) = \max\{\mathsf{depth}(p)+1, \mathsf{depth}(q)+1\}$$

$$\mathsf{depth}(\bigcirc^{\mathsf{u}} P) = \sup\{\mathsf{depth}(p')+1 \mid p' \in P\}\,.$$

The notion of structural depth lifts naturally to defining assignments (N, E). We let $\mathsf{depth}(E)$ be the smallest limit ordinal strictly greater than $\mathsf{depth}(E(A))$ for all $A \in N$. So, whenever the paper mentions the set of all terms $\mathbf{T}$, we refer, in fact, to restriction $\mathbf{T}_E = \{p \mid \mathsf{depth}(p) < \mathsf{depth}(E)\}$. The set of permitted contexts $\mathbf{C}_E$ is defined in the same way, as contexts are defined as terms built from $(N \uplus \{\bullet\}, E)$. We let $\mathbf{C}_E = \{c[\bullet] \mid \mathsf{depth}(c[\bullet]) < \mathsf{depth}(E)\}$, and drop the subscript from this set as well. This restriction indeed does not hinder our development. All definitions we apply to terms can be expressed as context free replacements and the set $\mathbf{T}_E$ is closed under such replacements. The only exceptions build Axioms (L1) and (LFP), for which we present side conditions so to stay in the restricted set $\mathbf{T}_E$.

Lemma 17: Let (N, E) be a defining assignment to h-terms. For all $p \in \mathbf{T}_E$ and all $c[\bullet] \in \mathbf{C}_E$, $c[p] \in \mathbf{T}_E$.

Proof: Proof is by an induction on the structure of $c[\bullet]$. Let $p \in \mathbf{T}_E$ and let $\alpha = \mathsf{depth}(E)$. Note that $\mathsf{depth}(p) < \alpha$. The case $c[\bullet] = p \in \mathbf{T}_E$ is clear, since $c[\bullet] = c[p]$. For the case $c[\bullet] = \bullet$, $\mathsf{depth}(c[p]) = \mathsf{depth}(p) < \alpha$.

For the concatenative inductive case, let $c[\bullet] = q.d[\bullet]$. The case $c[\bullet] = d[\bullet].q$ is analogous. Since $\mathsf{depth}(c[\bullet]) < \alpha$, we also have $\mathsf{depth}(d[\bullet]) < \alpha$ and $\mathsf{depth}(q) < \alpha$. Applying the induction hypothesis yields $\mathsf{depth}(d[p]) < \alpha$. And since α is a limit ordinal, also $\mathsf{depth}(q)+1 < \alpha$ and $\mathsf{depth}(c[p])+1 < \alpha$.

For the choice inductive case, let $c[\bullet] = \bigcirc^{\mathsf{u}}\{d[\bullet]\} \cup Q$. We have $\mathsf{depth}(c[p]) = \sup(\{\mathsf{depth}(d[p])+1\} \cup \{\mathsf{depth}(q) \mid q \in Q\})$. This is equal to the maximum of $\mathsf{depth}(d[p])+1$ and $\sup\{\mathsf{depth}(r)+1 \mid r \in Q\}$. We know that $\mathsf{depth}(c[\bullet]) < \alpha$. So, $\sup\{\mathsf{depth}(r)+1 \mid r \in Q\} \leq \mathsf{depth}(c[\bullet]) < \alpha$. Per definition, we also have $\mathsf{depth}(d[\bullet]) < \alpha$. We apply the induction hypothesis to get $\mathsf{depth}(d[p]) < \alpha$ and due to α being a limit ordinal, $\mathsf{depth}(c[p]) < \alpha$. ∎

An important implication of Lemma 17 is that the successor relation is well defined for $\mathbf{T}_E$. All rewriting has the form $c[\bigcirc^{\mathsf{u}} P] \to c[p]$ with $p \in P$ or $c[A] \to c[E(A)]$ for some context $c[\bullet]$ with $\mathsf{depth}(c[\bullet]) < \mathsf{depth}(E)$. Then, the successors have $\mathsf{depth}(c[p]) < \mathsf{depth}(c[\bigcirc^{\mathsf{u}} P]) < \mathsf{depth}(E)$ for all $p \in P$, and $\mathsf{depth}(E(A)) < \mathsf{depth}(E)$ for all $A \in N$.

1) Axiom Side Conditions on Depth: The restriction to $\mathbf{T}_E$ also restricts the axiom system to terms only in $\mathbf{T}_E$ This affects Axioms (L1) and (FP).

For Axiom (L1), note that $\bigcirc^{\mathsf{u}} \mathbf{T}_E \notin \mathbf{T}_E$.

$$\text{(L1)}\ \frac{\forall i \in I.\ p_i \sqsubseteq q_i}{\bigcirc^{\mathsf{u}}\{p_i \mid i \in I\} \sqsubseteq \bigcirc^{\mathsf{u}}\{q_i \mid i \in I\}}$$

So to keep terms in $\mathbf{T}_E$, we impose the following side condition to Axiom (L1): $\sup\{\mathsf{depth}(p_i)+1 \mid i \in I\} < \mathsf{depth}(E)$ and $\sup\{\mathsf{depth}(q_i)+1 \mid i \in I\} < \mathsf{depth}(E)$. This results in $\bigcirc^{\mathsf{u}}\{p_i \mid i \in I\} \in \mathbf{T}_E$ and $\bigcirc^{\mathsf{u}}\{q_i \mid i \in I\} \in \mathbf{T}_E$.

For Axiom (FP), note that $\mathbf{T}_E$ is closed under finite substitutions, but not under infinite substitutions. However, N is infinite, so for arbitrary p_N, it is not guaranteed that $E(A)\{N/p_N\} \in \mathbf{T}_E$.

$$\text{(LFP)}\ \frac{\forall A \in N.\ E(A)\{N/p_N\} \sqsubseteq p_A}{B \sqsubseteq p_B}$$

To ensure that the substituted term is part of $\mathbf{T}_E$, we require $\mathsf{depth}(E(A)\{N/p_N\}) < \mathsf{depth}(E)$ for all $A \in N$. In fact, the requirement is already implicitly stated by the axiom. The precondition requires $E(A)\{N/p_N\} \sqsubseteq p_B$ and $\sqsubseteq \subseteq \mathbf{T}_E \times \mathbf{T}_E$. So, $\mathsf{depth}(E(A)\{N/p_N\}) < \mathsf{depth}(E)$ is already required by the axiom implicitly.

B. Strategy Tree Bounds

Let $G = (V, v, \mathsf{own}, E)$ be a game arena with reachability objective $O \subset V$ and $\sigma : V \to V$ be a winning strategy for Eve from starting position $v \in V$. Consider the subgraph T of (V, E) reachable from v, where for all $v \in V$ owned by Eve only the successor $\sigma(v)$ is part of T. Since σ is a winning strategy, all paths from v in T must reach the objective in finitely many steps, i.e. T is a tree and every branch is finite. We prove in a second, that this is sufficient to obtain an ordinal α to bound the depth of T. If Eve has a strategy tree with depth α, we say that Eve wins in α turns.

The existence of uniform positional strategies results in an important property. If Eve wins from $v \in V$ in α turns, then there must be a $w \in E(v)$ from which Eve wins in $\gamma < \beta$ turns, if $\mathsf{own}(v) = Eve$. If $\mathsf{own}(v) = Adam$, Eve wins from all $w \in E(v)$ in $\alpha_w < \alpha$ turns.

It remains to prove that the absence of an infinite branch is sufficient to obtain α. Let $T = (V, E)$ be a directed tree and α_V be the smallest ordinal with $|V| \leq |\alpha_V|$.

Definition 6: A function $\mathsf{depth} : V \to \alpha_V$ is a depth assignment when $\mathsf{depth}(v) < \mathsf{depth}(w)$ for all $(v, w) \in E$. Note that $E(v) = \emptyset$ implies $\mathsf{depth}(v) = \sup(\emptyset) = 0$.

Lemma 18: If a directed tree $T = (V, E)$ has no infinite path then T has a depth assignment $\mathsf{depth} : V \to \alpha_V$.

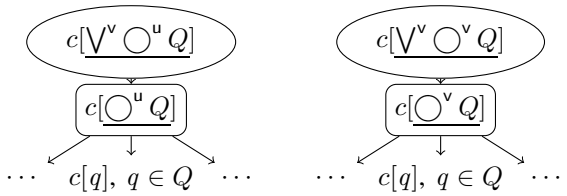
Proof: We prove the contraposition. Let $T = (V, E)$ be a directed tree with root $r \in V$. We use $T_v = (V_v, E_v)$ for the subtree rooted in $v \in V$. The key insight is that for all $v \in V$, where T_v has no depth assignment, there must be $w \in E_v(v)$ where also T_w has no depth assignment. To see the validity of this statement, suppose the existence of $v \in V$ where T_v has no depth assignments, while all $w \in E(v)$, T_w have depth assignments depth_w. Note that all V_w are disjoint. Then $\mathsf{depth} : V_v \to \alpha_{V_v}$ is a depth assignment, where $\mathsf{depth}(u) = \mathsf{depth}_w(u)$ if $u \in V_w$, and $\mathsf{depth}(v) = \sup\{\mathsf{depth}(w) + 1 \mid w \in E_v(v)\}$. The depth property is satisfied for all $u \in V_w$ and also for $v \in V_v$. So $\mathsf{depth} : V_v \to \alpha_{|V_v|}$ is a depth assignment, which contradicts the assumption.

Let $T = (V, E)$ have no depth assignment. We inductively construct an infinite sequence of nodes $(v_0, v_1, \ldots)$ with $v_i, v_{i+1} \in E$ and so that T_{v_i} has no depth assignment. The root is $v_0 = r$ and to extend $(v_0, ..., v_n)$, where T_{v_n} has no depth assignment, we choose any v_{n+1} such that $T_{v_{n+1}}$ has no depth assignment either, which we have shown to exist. ■

C. Soundness: Missing Axiom Proofs

Proofs of the remaining axioms: In all of the soundness proofs, we conclude with Lemma 2 to generalize from contexts for which at least one side of the conclusion is immediate, to all contexts. To avoid repetition, this conclusion is ommitted.

Axiom (N): We only show the case $\bigvee^{\mathsf{v}} \bigcirc^{\mathsf{u}} Q \simeq \bigvee^{\mathsf{v}} \bigcirc^{\mathsf{v}} Q$ with $\mathsf{v} < \mathsf{u}$. Like in the previous proof, both terms are immediate for the context $c[\bullet]$ of interest. The key is to note that the inner choice cannot be resolved until the outer choice has been made. The game arenas are:



As before, translation of strategies is straightforward.

Axiom (B1): To show $\bot \preceq p$, note that Eve never wins from a term $c[\bot]$ and hence the implication is trivial.

Axiom (L1): Assume $p_i \sqsubseteq q_i$ for all $i \in I$ for which also $p_i \preceq q_i$ holds. Further let $\bigcirc^{\mathsf{u}}\{p_i \mid i \in I\}$ and $\bigcirc^{\mathsf{u}}\{q_i \mid i \in I\}$ be valid terms (i.e. they satisfy the depth constraints from Appendix A). Acquire $c[\bullet]$ for which one of these terms are immediate. Since $\mathsf{urg}(\bigcirc^{\mathsf{u}}\{p_i \mid i \in I\}) = \mathsf{urg}(\bigcirc^{\mathsf{u}}\{q_i \mid i \in I\}) = \mathsf{u}$, Lemma 4 tells us that both terms are immediate for $c[\bullet]$. Fix an objective $O \subseteq \Sigma^*$. Let $c[\bigcirc^{\mathsf{u}}\{p_i \mid i \in I\}] \Downarrow O$. We have that $\mathsf{succ}(c[\bigcirc^{\mathsf{u}}\{p_i \mid i \in I\}]) = \{c[p_i] \mid i \in I\}$. Then, for some $i \in I$ ($\bigcirc = \vee$) [for all $i \in I$ ($\bigcirc = \wedge$)] holds $c[p_i] \Downarrow O$. Since $p_i \preceq q_i$, also $c[q_i] \Downarrow O$. Thus, $c[\bigcirc^{\mathsf{u}}\{p_i \mid i \in I\}] \Downarrow O$.

Axiom (L4): Let $\bigcirc^{\mathsf{u}}_{i \in I} \bigcirc^{\mathsf{u}} P_i$ and $\bigcirc^{\mathsf{u}} \bigcup_{i \in I} P_i$ be terms. Let $c[\bullet]$ be a context where one, and by Lemma 4 both, terms are immediate. After one move from $c[\bigcirc^{\mathsf{u}}_{i \in I} \bigcirc^{\mathsf{u}} P_i]$, the resulting term is always of the form $c[\bigcirc^{\mathsf{u}} P_i]$ for some $i \in I$. Since $\mathsf{urg}(\bigcirc^{\mathsf{u}} P_i) = \mathsf{u}$, Lemma 4 states that this term is also immediate for $c[\bullet]$. Then $\mathsf{succ}(c[\bigcirc^{\mathsf{u}} P_i]) = \{c[p] \mid p \in P_i\}$. This position is owned by the same player that owns the initial position, $c[\bigcirc^{\mathsf{u}}_{i \in I} \bigcirc^{\mathsf{u}} P_i]$ and the position $c[\bigcirc^{\mathsf{u}} \bigcup_{i \in I} P_i]$. Then, in two moves, this player reaches $\bigcup_{i \in I} \{c[p] \mid p \in P_i\} = \{c[p] \mid p \in \bigcup_{i \in I} P_i\}$ from $c[\bigcirc^{\mathsf{u}}_{i \in I} \bigcirc^{\mathsf{u}} P_i]$. We also have $\mathsf{succ}(c[\bigcirc^{\mathsf{u}} \bigcup_{i \in I} P_i]) = \{c[p] \mid p \in \bigcup_{i \in I} P_i\}$. So it is straightforward to lift the strategies from one term to the other easily under any objective.

Axiom (L5): Let $p, q \in \mathbf{T}$ with $\mathsf{urg}(p) \leq \mathsf{u}$. Acquire a context $c[\bullet]$ for which one of p or $p \vee^{\mathsf{u}} q$ is immediate. If p is immediate for $c[\bullet]$, since $\mathsf{urg}(p) \leq \mathsf{u} = \mathsf{urg}(p \vee^{\mathsf{u}} q)$, $p \vee^{\mathsf{u}} q$ is also immediate for $c[\bullet]$. Then, in any case, $p \vee^{\mathsf{u}} q$ is immediate for $c[\bullet]$. Fix an objective $O \subseteq \Sigma^*$ and let $c[p] \Downarrow O$. Since $c[p \vee^{\mathsf{u}} q]$, Eve can choose p in the inserted choice to reach $c[p]$ and win, i.e. $c[p \vee^{\mathsf{u}} q] \Downarrow O$.

Axiom (L3): Let $p, q \in \mathbf{T}$ and let $\mathsf{urg}(p) \leq \mathsf{u}$. We will only show $p \simeq p \vee^{\mathsf{u}} (p \wedge^{\mathsf{u}} q)$. The proof of the dual statement is analogous. Fix an objective $O \subseteq \Sigma^*$. Acquire a context $c[\bullet]$ for which one of p or $p \vee^{\mathsf{u}} (p \wedge^{\mathsf{u}} q)$ be immediate. Similarly to (L5), $p \vee^{\mathsf{u}} (p \wedge^{\mathsf{u}} q)$ is guaranteed to be immediate. Let $c[p] \Downarrow O$. Then Eve can play $c[p \vee^{\mathsf{u}} (p \wedge^{\mathsf{u}} q)] \to c[p]$ and win, so $c[p \vee^{\mathsf{u}} (p \wedge^{\mathsf{u}} q)] \Downarrow O$ as well. Let $c[p] \Downarrow O$ not hold. Then Adam has a winning strategy from $c[p]$, since reachability games are determined. If Eve were to play $c[p \vee^{\mathsf{u}} (p \wedge^{\mathsf{u}} q)] \to c[p]$, Adam would win from this position. If Eve were to instead play $c[p \vee^{\mathsf{u}} (p \wedge^{\mathsf{u}} q)] \to c[p \wedge^{\mathsf{u}} q]$, Adam can play $c[p \wedge^{\mathsf{u}} q] \to c[p]$ and win. Thus $c[p \vee^{\mathsf{u}} (p \wedge^{\mathsf{u}} q)] \Downarrow O$ does not hold.

Axiom (L2): Let $\bigwedge^{\mathsf{u}}_{i \in I} \bigvee^{\mathsf{u}} P_i \in \mathbf{T}$. Per Axiom of choice, $\{f \mid f : I \to P_I\} \neq \emptyset$ and $\bigvee^{\mathsf{u}}_{f : I \to P_I} \bigwedge^{\mathsf{u}} \{f(i) \mid i \in I\}$ is well defined. Let $\bigvee^{\mathsf{u}}_{f : I \to P_I} \bigwedge^{\mathsf{u}} \{f(i) \mid i \in I\} \in \mathbf{T}$. Fix an objective $O \subseteq \Sigma^*$ and acquire a context $c[\bullet]$ for which one of $\bigvee^{\mathsf{u}}_{f : I \to P_I} \bigwedge^{\mathsf{u}} \{f(i) \mid i \in I\}$ and $\bigwedge^{\mathsf{u}}_{i \in I} \bigvee^{\mathsf{u}} P_i$ is immediate. Since both terms have urgency u, Lemma 4 states that both terms are immediate. Per definition, we have $c[\bigwedge^{\mathsf{u}}_{i \in I} \bigvee^{\mathsf{u}} P_i] \Downarrow O$ if and only if for all $i \in I$, there is a $p \in P_i$ with $c[p] \Downarrow O$. Since we assume axiom of choice, we can apply Skolemization to get the following equivalent statement: There is a $f : I \to P_I$ where for all $i \in I$, $c[f(i)] \Downarrow O$. But this is equivalent to $c[\bigvee^{\mathsf{u}}_{f : I \to P_I} \bigwedge^{\mathsf{u}} \{f(i) \mid i \in I\}] \Downarrow O$. Then $c[\bigvee^{\mathsf{u}}_{f : I \to P_I} \bigwedge^{\mathsf{u}} \{f(i) \mid i \in I\}] \Downarrow O$ if and only if $c[\bigwedge^{\mathsf{u}}_{i \in I} \bigvee^{\mathsf{u}} P_i] \Downarrow O$.

Axiom (D2): Let $p, \bigcirc^{\mathsf{u}} Q \in \mathbf{T}$ where $\mathsf{urg}(p) \leq \mathsf{u}$. Acquire a context for which at least one, and per Lemma 4 both, of $(\bigcirc^{\mathsf{u}} Q).p$ and $\bigcirc^{\mathsf{u}}\{q.p \mid q \in Q\}$ are immediate. We have the owner $\mathsf{own}\left(c[(\bigcirc^{\mathsf{u}} Q).p]\right) = \mathsf{own}\left(c[\bigcirc^{\mathsf{u}}\{q.p \mid q \in Q\}]\right)$. It follows that we have $\mathsf{succ}(c[(\bigcirc^{\mathsf{u}} Q).p]) = \{c[q.p] \mid q \in Q\}$ due to $\mathsf{succ}((\bigcirc^{\mathsf{u}} Q).p) = \{q.p \mid q \in Q\}$. Further, $\mathsf{succ}(c[\bigcirc^{\mathsf{u}}\{q.p \mid q \in Q\}]) = \{c[q.p] \mid q \in Q\}$ as well. So under any objective $O \subseteq \Sigma^*$, lifting the strategies from one term to the other is straightforward.

Axiom (M): Let $w, v \in \mathbf{W}$ with $w \overset{*}{=} v$. Acquire a context $c[\bullet]$ that is immediate for one of w or v. Since these are both word terms, the only way one of these terms can be immediate

is if $c[\bullet]$ is a concatenation of terminals and $\bullet$. Then, we have $c[w] \stackrel{*}{=} c[v]$. For an objective $O \subseteq \Sigma^*$, we also see that $c[w] \Downarrow O$ if and only if $c[v] \Downarrow O$.

Axiom (S): Let $O \subseteq \Sigma^*$ be an objective. Let $w, v \in \Sigma^*_{\mathsf{err}}$ be word terms with $w \sqsubseteq_O v$ due to $w \leq^s_O v$. Let $c[\bullet]$ be a context for which one of the words is immediate. As the words have urgency zero, the context muss be a word as well: $c[\bullet] = x.\bullet.y$ (ignoring the bracketing) for $x, y \in \Sigma^*_{\mathsf{err}}$. Assume Eve wins O from $c[w] = x.w.y$. Then $x.w.y \in O$. By definition of $\leq^s_O$, we get $x.v.y \in O$. So Eve wins O from $c[v]$ as well.

Axiom (FP): Let $A \in N$. Acquire a context $c[\bullet]$ for which at least one of A or $E(A)$ is immediate. Since $\mathsf{urg}(A) = \mathsf{h} \geq E(A)$, Lemma 4 tells us that A is guaranteed to be immediate. The term $c[\underline{A}]$ has exactly one successor, $c[E(A)]$. Then under any objective $O \subseteq \Sigma^*$, $c[\underline{A}] \Downarrow O$ if and only if $c[E(A)] \Downarrow O$. ∎

D. Normalization

We provide the proof for Lemma 5. The function $\mathsf{nf}(p)$ is defined by induction. For p being a terminal or $\mathsf{skip}, \mathsf{err}$, we use (L6) to introduce a sequence of 2h choices over singleton sets and arrive at a aterm $\mathsf{nf}(p) \equiv p$ in normal form. For a concatenation or choice, we recursively normalize the operands and then invoke specialized functions that rely on the operands being normalized:

$$\begin{aligned} \mathsf{nf}(p.q) &= \mathsf{nfconc}(\mathsf{nf}(p).\mathsf{nf}(q)) \\ \mathsf{nf}(\bigcirc^{\mathsf{u}} P) &= \mathsf{nfchoice}(\bigcirc^{\mathsf{u}} \{\mathsf{nf}(p) \mid p \in P\}) \ . \end{aligned}$$

Lemma 19: Let $R \subseteq NF_{\mathsf{h}}$ and $r = \bigcirc^{\mathsf{u}} R$. We can find $\mathsf{nfchoice}(r) \in NF_{\mathsf{h}}$ with $\mathsf{nfchoice}(r) \equiv r$.

Proof: Consider the proof of Lemma 21. ∎

Lemma 20: For $p, q \in NF_{\mathsf{h}}$ we can find $\mathsf{nfconc}(p.q) \in NF_{\mathsf{h}}$ with $\mathsf{nfconc}(p.q) \equiv p.q$.

Proof: We strengthen the statement and show that for all urgencies u, if we have normal form terms $p, q \in NF_{\mathsf{u}}$, then we can obtain a normal form in NF_{u}. We proceed by induction on u. For the base case $\mathsf{u} = 0$, (M) yields $p.q \stackrel{*}{=} r \in NF_0$.

For the inductive case, let $\mathsf{u} > 0$. Then $p = \bigvee^{\mathsf{u}}_{i \in I} \bigwedge^{\mathsf{u}} P_i$ and $q = \bigvee^{\mathsf{u}}_{j \in J} \bigwedge^{\mathsf{u}} Q_j$. So we can write:

$$\begin{aligned} p.q &= (\bigvee^{\mathsf{u}}_{i \in I} \bigwedge^{\mathsf{u}} P_i).(\bigvee^{\mathsf{u}}_{j \in J} \bigwedge^{\mathsf{u}} Q_j) \\ &\stackrel{2\times(\mathrm{D2})}{\equiv} \bigvee^{\mathsf{u}}_{i \in I} \bigwedge^{\mathsf{u}}_{p' \in P_i} p'.(\bigvee^{\mathsf{u}}_{j \in J} \bigwedge^{\mathsf{u}} Q_j) \\ &\stackrel{2\times(\mathrm{D1})}{\equiv} \bigvee^{\mathsf{u}}_{i \in I} \bigwedge^{\mathsf{u}}_{p' \in P_i} \bigvee^{\mathsf{u}}_{j \in J} \bigwedge^{\mathsf{u}}_{q' \in Q_j} p'.q' \ . \end{aligned}$$

We have $p', q' \in NF_{\mathsf{u}-1}$. We apply the induction hypothesis to obtain $\mathsf{nfconc}(p'.q') \in NF_{\mathsf{u}-1}$

$$\stackrel{\text{I.H.}}{\equiv} \bigvee^{\mathsf{u}}_{i \in I} \bigwedge^{\mathsf{u}}_{p' \in P_i} \bigvee^{\mathsf{u}}_{j \in J} \bigwedge^{\mathsf{u}}_{q' \in Q_j} \mathsf{nfconc}(p'.q') \ .$$

The term is not in normal form due to the two layers of u choices. We apply Lemma 19 to obtain a normal form. ∎

E. Completeness: Characteristic Context

The missing inductive case is $p = \bigwedge^{\mathsf{u}} P \in ANF_{\mathsf{u}}$. We set

$$t_p \quad = \quad \bigvee^{\mathsf{u}} \{t_r \mid r \in P\}.$$

Since p is not minimal, no element $r \in P$ is minimal and hence the t_r are guaranteed to exist. To see Equivalence (1), consider $\bigwedge^{\mathsf{u}} Q \in ANF_{\mathsf{u}}$:

$$\begin{aligned} & (\bigwedge^{\mathsf{u}} Q).(\bigvee^{\mathsf{u}} \{t_r \mid r \in P\}) \Downarrow O \\ \text{iff} \quad & \forall s \in Q.\ \exists r \in P.\ s.t_r \Downarrow O \\ \{\text{I.H.}\} \quad \text{iff} \quad & \forall s \in Q.\ \exists r \in P.\ r \subseteq_O s. \\ \text{iff} \quad & \bigwedge^{\mathsf{u}} P \ \subseteq_O \ \bigwedge^{\mathsf{u}} Q. \end{aligned}$$

F. Completeness: Domain Shattering Objectives

Let $O = \{a^{(n^2)} \mid n \in \mathbb{N}\}$ if $\Sigma = \{a\}$ and $O = \{w.w^{reverse} \mid w \in \Sigma^*\}$ if $|\Sigma| > 1$. We show that for all $w, v \in \mathbf{W}$ with $w \stackrel{*}{\neq} v$, we can find a $x \in \Sigma^+$ with $w.x \in O$ and $v.x \notin O$. This proves that O is right seperating. In particular, it also showcases that Axiom (S) cannot relate terms w, v unless $w \stackrel{*}{=} v$. So it also proves that O is domain-shattering. This also shows for all $w \in \Sigma^+$ the existence of x with $w.x \in O$ to establish the right seperability of w and $\bot$.

Case $|\Sigma| = 1$: Let $w = a^i$ and $v = a^j$ with $i \neq j$. Then set $x = a^{t^2 - i}$ with $t = i + j + 1$. Obviously, $w.x \in O$. Suppose $a^j.a^{t^2-i} \in O$. Then, $j + t^2 - i = k^2$ for some $k \in \mathbb{N}$. So $j - i = t^2 - k^2 = (t - k) \cdot (t + k)$. $i \neq j$ implies $|t - k| \geq 1$. Note that $k, i, j \geq 0$ so we get the contradiction $|j - i| < i + j < t + k < |t - k| \cdot |t - k| = |j - i|$.

Case $|\Sigma| > 1$: Let $w = w_0 w_1 \dots w_n$ and $v = v_0 v_1 \dots v_m$. Let $a \neq w_m$ be a terminal. We set $x = a.a.w^{reverse}$, where $a \neq w_{|v|}$ if $|w| > |v|$, $a \neq v_{|w|}$ if $|v| > |w|$, and a arbitrary if $|w| = |v|$. Suppose $v.a.a.w^{reverse} \in O$. Then, $v.a.a.w^{reverse} = w.a.a.v^{reverse}$. If $|w| = |v|$, $w = v$ is a contradiction. Otherwise, if $|v| < |w|$, we have $(v.a)_{|v|} = a \neq w_{|v|} = (w.a)_{|v|}$ for a contradiction. Similarly, if $|w| < |v|$, the contradiction is $(w.a)_{|w|} = a \neq v_{|w|} = (v.a)_{|w|}$.

G. Specialized Normal Form

Before we address the normalization algorithm, we take a brief detour to prove the utilized proof rules correct. We used (or will use) the following proof rules, which follow from the axioms in Figure 2.

$$\text{(REP)} \ \frac{\forall A \in N.\ p_A \sqsubseteq A}{q\{N/p_N\} \sqsubseteq q} \qquad \text{(L6)} \ \frac{\mathsf{urg}(p) \leq \mathsf{u}}{p \equiv \bigcirc^{\mathsf{u}} p}$$

$$\text{(DN)} \ \frac{\mathsf{v} < \mathsf{u}}{\bigwedge^{\mathsf{v}} \bigcirc^{\mathsf{u}} Q \equiv \bigwedge^{\mathsf{v}} \bigcirc^{\mathsf{v}} Q}$$

Proof sketch: For (REP), we use (L2) and the congruence rule inductively on the subterms. Utilizing (L3) twice yields (L6). G follows from

$$\bigwedge_{i\in I}^{\mathsf{v}} \bigcirc^{\mathsf{u}} Q_i \overset{\text{(L6)}}{\equiv} \bigvee^{\mathsf{v}} \bigwedge_{i\in I}^{\mathsf{v}} \bigcirc^{\mathsf{u}} Q_i \overset{\text{(L2)}}{\equiv} \bigwedge_{i\in I}^{\mathsf{v}} \bigvee^{\mathsf{v}} \bigcirc^{\mathsf{u}} Q_i \overset{\text{(N)}}{\equiv} \bigwedge_{i\in I}^{\mathsf{v}} \bigvee^{\mathsf{v}} \bigcirc^{\mathsf{v}} Q_i \overset{\text{(L6)}}{\equiv} \bigwedge_{i\in I}^{\mathsf{v}} \bigcirc^{\mathsf{v}} Q_i \quad \blacksquare$$

We now provide a sketch of the normalization algorithm with the complexity $(|p| + |E||N|) \cdot \exp^O_{2\mathsf{h}-1}$ [1] as given in Lemma 9.

We use methods from Section V with slight modifications. To avoid repetition, we only note the modifications that need to be made, instead of giving a full algorithm. The section takes the form of constructing modified normalization functions $\mathsf{nf}_O(.)$, $\mathsf{nfchoice}_O(.)$, and $\mathsf{nfconc}_O(.)$. We provide lemmas that reference those in Section V. The $\mathsf{nfchoice}_O(.)$ and $\mathsf{nfconc}_O(.)$ calls made by $\mathsf{nf}_O(.)$ are kept the same. Note that implementing $\mathsf{nfchoice}_O(.)$ and $\mathsf{nfconc}_O(.)$ with time complexity $|p| \cdot \exp^O_{2\mathsf{h}-1}$ for the input term p means that $\mathsf{nf}_O(q)$ is also constructed in $|q| \cdot \exp^O_{2\mathsf{h}-1}$ time, if q has no non-terminals. For the rest of the section, fix a regular objective $O \subseteq \Sigma^*$, a finite set of non-terminal symbols N, and a finitary defining assignment $E : N \to \mathbf{T}$.

Lemma 21 (Modifies Lemma 19): Let $P \subseteq SNF^O_{\mathsf{h}}$ and $r = \bigcirc^{\mathsf{u}} P$. We can find $\mathsf{nfchoice}_O(r) \in SNF^O_{\mathsf{h}}$ with $\mathsf{nfchoice}_O(r) \equiv_O r$ in $|r| \cdot \exp^O_{2\mathsf{h}-1}$ time.

Proof: By induction on h. Note that for normalforms in SNF^O_{u} also have maximal urgency u. In any case of u we need to normalize a term $r = \bigcirc^{\mathsf{u}}_{i\in I} \bigvee^{\mathsf{h}} P_i$. The harder case is $\bigcirc = \bigwedge$. We apply (N) if $\mathsf{u} < \mathsf{h}$ and (L2) to obtain $\bigvee^{\mathsf{u}}_{f:I\to P_I} \bigwedge^{\mathsf{u}}_{i\in I} f(i)$ and utilize (L4) to combine the $\bigwedge^{\mathsf{u}}_{i\in I}$ with the $f(i) \in SANF^O_{\mathsf{u}}$:

$$\mathsf{nfchoice}_O(r) = \bigvee_{f:I\to P_I}^{\mathsf{u}} \bigwedge^{\mathsf{u}} Q_f$$

where $Q_f = \bigcup\{Q \mid f(i) = \bigwedge^{\mathsf{u}} Q, i \in I\}$. Naively, applying Axiom (L2) requires us to enumerate all the choice functions $f : I \to P_I$. However, we do not need to account for all of them, because after applying (L4) for a single f, we know that $\bigwedge^{\mathsf{u}} Q_f \in SANF^O_{\mathsf{u}}$. Knowing this means that the distribution considers way more functions f (namely $\prod_{i\in I} |P_i| \geq 2^{|SNF^O_{\mathsf{u}}|}$) than there can be sets Q_f ($|SANF^O_{\mathsf{u}}|$). Instead, we can use (L4) to split the application of (L2) into $|I| - 1$ many single applications of (L2) and keeping the size of the intermediary results bound by $|SANF^O_{\mathsf{u}}| = \exp^O_{2\mathsf{u}-1}$: In the case of $J \subseteq I$ with $|J| = 2$, i.e. constant, the number of functions $f : J \to P_J$ is bound by $|P_1| \cdot |P_2| \leq |SANF^O_{\mathsf{u}}|^2$. For such an f (with binary co-domain) the union Q_f can be computed in time $|SNF^O_{\mathsf{u}-1}|^2$ as long as the terms $f(j)$ are in normalform $f(j) \in SANF^O_{\mathsf{u}}$. In case of $\mathsf{h} = \mathsf{u}$, this will be the case. Otherwise, $\mathsf{u} < \mathsf{h}$ and we apply the induction hypothesis to obtain normalforms equivalent to $f(1)$ and $f(2)$ before computing Q_f. That way, we compute no more than $|I| \leq |r|$ many (L4) for splitting, $|I|$ many (L2) enumerating $|SANF^O_{\mathsf{u}-1}|^2 \leq \exp^O_{2\mathsf{u}-1}$ functions f each, and for each f we apply (L4) to create the union Q_f in time $|SNF^O_{\mathsf{u}}|^2 \leq \exp^O_{2\mathsf{u}-2}$. Together (for $\mathsf{h} = \mathsf{u}$):

$$\underbrace{|I|}_{\text{(L4)}} + |I| \cdot \underbrace{|SANF^O_{\mathsf{u}}|^2}_{\text{(L2)}} \cdot \underbrace{|SNF^O_{\mathsf{u}-1}|^2}_{\text{(L4)},Q_f} \leq |r| \cdot \exp^O_{2\mathsf{u}-1}$$

In case of $\mathsf{u} < \mathsf{h}$, the right summand changes to

$$|I| \cdot \underbrace{|SANF^O_{\mathsf{u}}|^2}_{\text{(L2)}} \cdot (\underbrace{|SNF^O_{\mathsf{u}-1}|^2}_{\text{(L4)},Q_f} + \underbrace{2|SNF^O_{\mathsf{h}-1}| \cdot \exp^O_{2\mathsf{h}-3}}_{\text{I.H.}}) \leq |r| \cdot \exp^O_{2\mathsf{h}-1}$$

Adding singleton choice operators (L6) to reobtain a term in SNF^O_{h} costs close to no time. $\blacksquare$

Lemma 22 (Modifies Lemma 20): For any $p, q \in SNF^O_{\mathsf{h}}$, we can find $\mathsf{nfconc}_O(p.q) \in SNF^O_{\mathsf{h}}$ with $\mathsf{nfconc}_O(p.q) \equiv_O p.q$ in $\exp^O_{2\mathsf{h}-1}$ time.

Proof: The proof in Section V proceeds by an induction on urgency. We change the base case to $\mathsf{h} = 1$ and for both, base and inductive case, apply (D1) and (D2) as in the proof of Lemma 20. The application of (D1) creates at most $|p|$ copies of q, and (D2) creates another $|q|$ copies of each p'. Call the resulting term s with $|s| \leq 2 \cdot |p| \cdot |q|$.

In the base case, p' and q' belong to SNF^O_0. We employ (FP) and find an $x \in SNF^O_0$ with $x \cong^s_O w.v$. Finding the syntactical congruence class of $w.v$ can be done in time $(\exp^O_0)^k$ for some fixed $k \in \mathbb{N}$ by naively checking.

In the inductive case, we apply the induction hypothesis to normalize the concatenative subterms $p'.q'$ into SNF^O_{h}. A term in SNF^O_{h} term is bound by size $|SANF^O_{\mathsf{h}}| \leq \exp^O_{2\mathsf{h}-1}$. After applying the induction hypothesis to each pair in s, the resulting term t has size $|t| \leq 2 \cdot |p| \cdot |q| \cdot \exp^O_{2\mathsf{h}-1} = \exp^O_{2\mathsf{h}-1}$. In Section V the layers of choices are resolved by invoking Lemma 19. In our case, we employ the modified Lemma 21 for another $|t| \cdot \exp^O_{2\mathsf{h}-1} \leq (\exp^O_{2\mathsf{h}-1})^2 = \exp^O_{2\mathsf{h}-1}$. $\blacksquare$

Lemma 23 (Modifies Lemma 6): For any $A \in N$, we can find $\mathsf{nf}(A) \in SNF^O_{\mathsf{h}}$ with $\mathsf{nf}(A) \equiv_O A$ in $|E| \cdot |N| \cdot \exp^O_{2\mathsf{h}-1}$ time.

Proof Sketch: We construct an increasing chain of SNF^O_{h} terms using the same least fixed point construction from Section V. The constructed term for component A has size at most $|E(A)| \cdot (\exp^O_{2\mathsf{h}-1})^2$. Per Lemmas 21 and 22, this term is normalized in $|E(A)| \cdot (\exp^O_{2\mathsf{h}-1})^3 = |E(A)| \cdot \exp^O_{2\mathsf{h}-1}$ time. So normalizing all components takes $|E| \cdot \exp^O_{2\mathsf{h}-1}$ time. The chain converges in at most $|N| \cdot |SANF^O_{\mathsf{h}}| = |N| \cdot \exp^O_{2\mathsf{h}-1}$ iterations. So the normalization process takes $|E| \cdot |N| \cdot \exp^O_{2\mathsf{h}-1}$ time. $\blacksquare$

We complete the normalization function analogously to Section V. To calculate $\mathsf{nf}_O(p)$ for any finitary $p \in \mathbf{T}$ with non-terminals, the algorithm first constructs $\mathsf{nf}_O(A)$ for all $A \in N$. This takes $|E| \cdot |N| \cdot \exp^O_{2\mathsf{h}-1}$ time. Then, we

[1] We use $\exp^O_{\mathsf{u}}$ for $\exp_{\mathsf{u}}(\mathcal{O}(|SNF^O_0|))$. Usages of $\leq$ and $=$ are to be understood by means of $\in$ or $\subseteq$.

let $\mathsf{nf}_O(p) = \mathsf{nf}_O(p\{N/p_N\})$ where $p_A = \mathsf{nf}_O(A)$. The normalization takes $|p| \cdot \exp^O_{2\mathsf{h}-1}$ time per Lemmas 21 and 22. This concludes the construction of the algorithm and thus proves Lemma 9.

H. Decidability and Complexity

Proof of Proposition 5: Consider two context free grammars $G_i = (N_i, E_i, S_i), i \in \{0, 1\}$ over some alphabet Σ and with disjoint sets of non-terminals $N_1 \cap N_2 = \emptyset$. Note that each non-terminal has a single production rule of the shape $E_i(A) = w_1 \mid \ldots \mid w_n$, $w_i \in (N \cup \Sigma)^*$. We reproduce this non-deterministic structure by yielding the choice to Eve. We construct the program-term grammar $(N_1 \cup N_2, E)$, where E is defined via $E(A) = \bigvee^1\{w_1, \ldots, w_n\}$ for each production rule of above shape. Kleene iteration yields the normal form

$$\mathsf{nf}(S_i) = \bigvee_1\{\bigwedge_1 w \mid w \in L(G_i)\} \simeq \bigvee_1 L(G_i)\,.$$

To finish, we utilize $\preceq_O \;=\; \preceq$ by choosing the shattering objective $O = \{w.w^{reverse} \mid w \in \Sigma^*\}$. Since $\preceq_O$ is the reflexivity relation on $\Sigma^* \times \Sigma^*$, the domination pre-order yields $\bigvee_1 L =_O \bigvee_1 M$ if and only if $L = M$ for any $L, M \subseteq \Sigma^*$. Using Lemma 8, Theorem 2, and that O is shattering and right-separating, $L(G_1) = L(G_2)$ if and only if $S_1 \simeq \bigvee_1 L(G_1) \simeq \bigvee_1 L(G_2) \simeq S_2$. ∎

Proof of Theorem 3: We show that h-DEC-$\preceq_O$ is PTIME-hard (wrt. log-space reductions) by sketching out a reduction from the Monotonic Circuit Value Problem [39] The problem consists of assigments of boolean variables $\mathsf{P}_{i\leq n}$ to $\vee/\wedge$ clauses built out of the variables $\mathsf{P}_{j<i}$, to true , or to false. The input is accepted if and only if the variable P_n evaluates to true. Let non-empty $O \subseteq \Sigma^*$ and let $w \in O$. For each boolean variable P_i, the log-space Turing Machine outputs a non-terminal P_i and a defining equation $E(P_i)$. The machine outputs $E(P_i) = w$ if $\mathsf{P}_i = \mathsf{true}$, and it outputs $E(P_i) = \mathsf{err}$ if $\mathsf{P}_i = \mathsf{false}$. For $\mathsf{P}_i = \mathsf{P}_{j_0} \wedge \ldots \wedge \mathsf{P}_{j_k}$, the machine outputs $E(P) = \bigwedge^1\{P_{j_l} \mid l \leq k\}$. For $\mathsf{P}_i = \mathsf{P}_{j_0} \vee \ldots \vee \mathsf{P}_{j_k}$, it outputs $\bigvee^1\{P_{j_l} \mid l \leq k\}$. The reduced problem instance asks whether $w \preceq_O P_n$. By induction on i we show $P_i \equiv_O w$ if P_i evaluates to true, and $P_i \equiv_O w$ and if P_i evaluates to false. Since $\equiv_O$ sound, and $w \not\preceq_O \mathsf{err}$ (with the witnessing context $c[\bullet] = \bullet$) this proof suffices.

For the base case, we have $\mathsf{P}_0 = \mathsf{true}$ or $\mathsf{P}_0 = \mathsf{false}$. After applying Axiom (FP), we get $P \equiv_O w$ and $P \equiv_O \mathsf{err}$ respectively. For the inductive case, we have $\mathsf{P}_i = \mathsf{true}$, $\mathsf{P}_i = \mathsf{false}$, $\mathsf{P}_i = \mathsf{P}_{j_0} \wedge \ldots \wedge \mathsf{P}_{j_k}$, or $\mathsf{P}_i = \mathsf{P}_{j_0} \vee \ldots \vee \mathsf{P}_{j_k}$. The first two subcases are already handled in the base case. The latter two cases are dual, so we only handle the $\wedge$-case. Per Axiom (FP), we have $P_i \equiv_O \bigwedge^1\{P_{j_l} \mid l \leq k\}$. The variable P_i evaluates to true, if and only if P_{j_l} evaluates to true for all $l \leq k$. Let all variables evaluate to true. Induction hypothesis delivers us $\mathsf{P}_{j_l} \equiv_O w$ and thus $P_i \equiv_O \bigwedge^1\{w\}$. The rule Axiom (L6) tells us $\bigwedge^1\{w\} \equiv_O w$. Wlog. let a variable P_{j_0} evaluate to false. then induction hypothesis delivers us $P_{j_0} \equiv_O \mathsf{err}$ and $P_{j_l} \equiv_O w$ or $P_{j_l} \equiv_O \mathsf{err}$ for all $0 < l \leq k$. We have $P_i \equiv_O \bigwedge^1\{\mathsf{err}\}$ or $P_i \equiv_O \bigwedge^1\{\mathsf{err}, w\}$. In the former case we have $P_i \equiv_O \mathsf{err}$ per Axiom (L6). In the latter case, the lattice axioms tell us that $\bigwedge^1\{w, \mathsf{err}\} \equiv_O \mathsf{err}$, and thus $P_i \equiv_O \mathsf{err}$. ∎

I. Upper Bound

In this section, we handle the omitted proofs from Section VIII. Namely, we prove Proposition 7, Lemma 10, the special case $\mathsf{h} = 1$ in Lemma 15 and we justify Equation (3).

Reasoning for Equation (3): To see Equation (3), note that by Equation (2) the characteristic term $\chi(c[\bullet])$ is an urgency-h choice owned by Adam over the set of $SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}$ solutions of $c[\bullet]$. This set of solutions is the union of the $SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}$ solutions for $s.\bullet.t$ with $t \in T$. The reason is that the choice over T in the context has a higher urgency than the inserted term. We can thus stratify Adam's choice into a choice over $t \in T$ followed by a choice of the $SNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}$ solutions for $s.\bullet.t$. Again by Equation (2), this is precisely the right-hand side of Equation (3).

Proofs of Lemma 10 and Proposition 7: Proofs of these statements require us to observe the inner workings of terms. To do so cleanly, we extend the relation $\overset{*}{=}$ to terms. The relation $\overset{*}{=}$ on terms is the smallest equivalence relation that contains $\overset{*}{=}$ on $\mathbf{W}$, and the equalities $p.(q.r) \overset{*}{=} (p.q).r$, $p.\mathsf{skip} \overset{*}{=} p$, $\mathsf{skip}.p \overset{*}{=} p$, $p.\mathsf{err}.q \overset{*}{=} \mathsf{err}$ for all $p, q, r \in \mathbf{T}$. Note that this is diffrent than Axiom (M), which allows us to apply $\overset{*}{=}$ to words enclosed by arbitrary contexts. We also define $\mathsf{hd} : \mathbf{T} \to \mathbf{W}$ and $\mathsf{tl} : \mathbf{T} \to \mathbf{T}$. We let $\mathsf{hd}(p) = w \in \mathbf{W}$ be the concatenation of outermost terminals (including any skip and err) that appear in p before the leftmost outermost action. If there are no such terms, $\mathsf{hd}(p) = \mathsf{skip}$. We let $\mathsf{tl}(p) = q$ be the concatenation of the remaining outermost actions and commands. If no outmost action exists in the term, we let $\mathsf{tl}(p) = \mathsf{skip}$. Note that $p \overset{*}{=} \mathsf{hd}(p).\mathsf{tl}(p)$ for all $p \in \mathbf{T}$. Finally, we define a set of quasi-runs QRun_w for all $w \in \Sigma^* \cup \{\mathsf{err}\}$. We employ the notation QRun_w for a word term $w \in \mathbf{W}$, as a shorthand for QRun_v where v is the monoid element from $\Sigma^* \cup \{\mathsf{err}\}$. We let $\mathsf{QRun}_{\mathsf{skip}} = \{\mathsf{skip}\}$ and $\mathsf{QRun}_{\mathsf{err}} = \{\mathsf{err}\}$. For a word $w = a_0 \ldots a_{k-1} \in \Sigma^+$, the set QRun_w contains all terms of the form $(i, p_1).(p_1, p_2) \ldots (p_{n-1}, p_n).\mathsf{tr}(a_n) \ldots \mathsf{tr}(a_{k-1})$, where $i \overset{a_0}{\to} p_1 \ldots p_{n-1} \overset{a_{n-1}}{\to} p_n$ is a run in the DFA for O. So an element of QRun_w runs the DFA on the prefix of w up to n and has undetermined transitions in form of the terms $\mathsf{tr}(a_i)$ for the remainder of w.

We find it useful to prove a stronger version of Lemma 10.

Lemma 24: Let $p \in \mathbf{T}$, $r \in \mathsf{QRun}_{\mathsf{hd}(p)}$ and $O \subseteq \Sigma^*$. Then for all $q \overset{*}{=} r.\mathsf{tr}(\mathsf{tl}(p))$, $p \Downarrow O$ if and only if $q \Downarrow \mathsf{tr}(O)$.

Assuming Lemma 24 we show Proposition 7.

Proof of Proposition 7: Let $O \subseteq \Sigma^*$. Per Corollary 2, we know that $p \preceq_O q$ holds if and only if $s.p.t \Downarrow O$ implies $s.q.t \Downarrow O$ for all $s \in SNF^O_{\mathsf{h}-1}$ and $t \in SNF^O_{\mathsf{h}}$. We apply Lemma 10 to see that this is equivalent to the statement $\mathsf{tr}(s.p.t) \Downarrow \mathsf{tr}(O)$ implies $\mathsf{tr}(s.q.t) \Downarrow \mathsf{tr}(O)$ for all $s \in SNF^O_{\mathsf{h}-1}$ and $t \in SNF^O_{\mathsf{h}}$. Per definition, we have $\mathsf{tr}(s.p.t) = \mathsf{tr}(s).\mathsf{tr}(p).\mathsf{tr}(t)$ and $\mathsf{tr}(s.q.t) = \mathsf{tr}(s).\mathsf{tr}(q).\mathsf{tr}(t)$.

For the moment, assume $TNF^{\mathsf{tr}(O)}_{\mathsf{u}} = \{\mathsf{tr}(r) \mid r \in SNF^O_{\mathsf{u}}\}$ (up to $\equiv_{\mathsf{tr}(O)}$) for all $\mathsf{u} \geq 1$ without proof. For $\mathsf{h} > 1$, this makes the previous statement equivalent with the desired $s.\mathsf{tr}(p).t \Downarrow O$ implies $s.\mathsf{tr}(p).t \Downarrow O$ for all $s \in TNF^{\mathsf{tr}(O)}_{\mathsf{h}-1}$ and $t \in TNF^{\mathsf{tr}(O)}_{\mathsf{h}}$.

Now let $\mathsf{h} = 1$. We show that for all $s \in SNF^O_0$ and $t \in SNF^O_1$, there are $s' \in TNF^{\mathsf{tr}(O)}_0$ and $t' \in TNF^{\mathsf{tr}(O)}_1$ where $s.r.t \Downarrow O$ if and only if $s'.\mathsf{tr}(r).t' \Downarrow \mathsf{tr}(O)$ for all $r \in SNF^O_1$. We can also conversely find $s \in SNF^O_0$ and $t \in SNF^O_1$ for all $s' \in TNF^{\mathsf{tr}(O)}_0$ and $t' \in TNF^{\mathsf{tr}(O)}_1$ with the same property. Then, the statement $s.p.t \Downarrow O$ implies $s.q.t \Downarrow O$ for all $s \in SNF^O_0$, $t \in SNF^O_1$ is equivalent to the statement $s'.\mathsf{tr}(p).t' \Downarrow \mathsf{tr}(O)$ implies $s'.\mathsf{tr}(q).t' \Downarrow \mathsf{tr}(O)$ for all $s' \in TNF^{\mathsf{tr}(O)}_0$, $t' \in TNF^{\mathsf{tr}(O)}_1$. Note that, we have $\mathsf{hd}(s.p.t) = s$ and $\mathsf{tl}(s.p.t) = p.t$. Per Lemma 24, we know that for all $r_s \in \mathsf{QRun}_s$, $s.p.t \Downarrow O$ if and only if $r_s.\mathsf{tr}(p.t) \Downarrow O$. We can let $s' = r_s = (i, p_1)(p_1, p_2)\dots(p_{n-1}, p_n) \in TNF^{\mathsf{tr}(O)}_0 \cap \mathsf{QRun}_s$ where the DFA runs on s from i to p_n and $t' = \mathsf{tr}(t)$.

Finally, we show our assumption $TNF^{\mathsf{tr}(O)}_{\mathsf{u}} = \{\mathsf{tr}(r) \mid r \in SNF^O_{\mathsf{u}}\}$ (up to $\equiv_{\mathsf{tr}(O)}$) for all $\mathsf{u} \geq 1$. The inclusion $TNF^{\mathsf{tr}(O)}_{\mathsf{u}} \subseteq \{\mathsf{tr}(r) \mid r \in SNF^O_{\mathsf{u}}\}$ follows from the fact that $\mathsf{tr}(\bigcirc^{\mathsf{u}} P) = \bigcirc^{\mathsf{u}}\{\mathsf{tr}(p) \mid p \in P\}$ along with the definitions of $TNF^{\mathsf{tr}(O)}_{\mathsf{u}}$ and $TANF^{\mathsf{tr}(O)}_{\mathsf{u}}$. The inclusion $\{\mathsf{tr}(r) \mid r \in SNF^O_{\mathsf{u}}\} \subseteq TNF^{\mathsf{tr}(O)}_{\mathsf{u}}$ is proven by induction on u. For the base case, we have $TNF^{\mathsf{tr}(O)}_1 = \{\mathsf{nf}(\mathsf{tr}(r)) \mid r \in SNF^O_1\}$ per definition. For the inductive case, the cases SNF^O_{u} and $SANF^O_{\mathsf{u}}$ are analogous, so we only handle one. Letting $\bigvee^{\mathsf{u}} P \in SNF^O_{\mathsf{u}}$, we get $\mathsf{nf}(\mathsf{tr}(\bigvee^{\mathsf{u}} P)) = \mathsf{nf}(\bigvee^{\mathsf{u}}\{\mathsf{nf}(\mathsf{tr}(r))\} \mid r \in P)$. Induction hypothesis tells us that $\mathsf{nf}(\mathsf{tr}(r)) \in TANF^{\mathsf{tr}(O)}_{\mathsf{u}}$ for all $r \in P$, so $\bigvee^{\mathsf{u}}\{\mathsf{nf}(\mathsf{tr}(r))\} \in TNF^{\mathsf{tr}(O)}_{\mathsf{u}}$. ■

The proof of Lemma 24 is more involved and relies on the following Lemmas 25 and 26, which we prove first.

Lemma 25: Let $p, q \in \mathbf{T}$ with $p \overset{*}{=} q$. Then $\mathsf{own}(p) = \mathsf{own}(q)$ and Eve wins from p in β moves if and only if Eve wins from q in β moves.

Proof Sketch: The definition of ownership already implies $\mathsf{own}(p) = \mathsf{own}(q)$ for all $p, q \in \mathbf{T}$ with $p \overset{*}{=} q$. The remainder of the statement is proven by transfinite induction on β. The base case follows from the monoid evaluation of word terms. We sketch out the inductive case. Wlog. we can only handle one direction of the implication. Let $p \overset{*}{=} q$ and let Eve win from p in β moves. In both terms, the same i-th concatenation operand (ignoring the bracketing) will be leading. Then, the successor sets are equal up to $\overset{*}{=}$. If $\mathsf{own}(p) = \mathsf{own}(q) = Eve$, then Eve wins from at least one $p' \in \mathsf{succ}(p)$ in $\gamma < \beta$ moves. Here, we can apply the induction hypothesis to lift the strategy to q. The case $\mathsf{own}(p) = \mathsf{own}(q) = Adam$ is dual. ■

We cal $p \in \mathbf{T}$ *headless* if $\mathsf{hd}(p) = \mathsf{skip}$. For headless terms, $\mathsf{tr}(.)$ and $\mathsf{succ}(.)$ commute.

Lemma 26: For headless $p \in \mathbf{T}$, $\mathsf{succ}(\mathsf{tr}(p)) = \mathsf{tr}(\mathsf{succ}(p))$.

Proof: The proof is by (transfinite) structural induction on $p \in \mathbf{T}$. For the base case, we let $p = A \in N$. This is the only base case, because p is only headless if $p = A \in N$. Per definition, we have $\mathsf{succ}(\mathsf{tr}(A)) = \{\mathsf{tr}(E(A))\} = \mathsf{tr}(\mathsf{succ}(A))$.

The first inductive case is $p = \bigcirc^{\mathsf{u}} P$. Per definition, $\mathsf{tr}(\bigcirc^{\mathsf{u}} P) = \bigcirc^{\mathsf{u}}\{\mathsf{tr}(p) \mid p \in P\}$. And thus, $\mathsf{succ}(\bigcirc^{\mathsf{u}}\{\mathsf{tr}(p) \mid p \in P\}) = \{\mathsf{tr}(p) \mid p \in P\} = \mathsf{tr}(\mathsf{succ}(\bigcirc^{\mathsf{u}} P))$.

The second inductive case is $p = q.r$. If q is not headless, then p would not be headless. So we deduce that q must be headless. A headless term must contain an outermost choice or a non-terminal, since terms can not be empty. So, $\mathsf{urg}(q) \geq 1$. We observe that for all $s \in \mathbf{T}$, $\mathsf{urg}(s) > 0$ implies $\mathsf{urg}(\mathsf{tr}(s)) = \mathsf{urg}(s)$ and $\mathsf{urg}(s) = 0$ implies $\mathsf{urg}(\mathsf{tr}(s)) \leq 1$. This is clear from the construction of $\mathsf{tr}(.)$: Only the urgencies of urgency 0 subterms change. These get replaced by urgency 1 terms, unless they are err or skip.

The first case is $\underline{q}.r$. So, we have $\mathsf{urg}(q) \geq \mathsf{urg}(r)$. When $\mathsf{urg}(r) = 0$, then $\overline{\mathsf{urg}}(\mathsf{tr}(q)) \geq 1 \geq \mathsf{urg}(\mathsf{tr}(r))$. Otherwise for q to be leading, $\mathsf{urg}(q) \geq 1$ must hold and

$$\mathsf{urg}(\mathsf{tr}(q)) = \mathsf{urg}(q) \geq \mathsf{urg}(r) = \mathsf{urg}(\mathsf{tr}(r))\,.$$

In either case we have $\underline{\mathsf{tr}(q)}.\mathsf{tr}(r)$. By induction hypothesis, $\mathsf{succ}(\mathsf{tr}(q)) = \mathsf{tr}(\mathsf{succ}(\overline{q}))$. This results in $\mathsf{succ}(\mathsf{tr}(q.r)) = \mathsf{succ}(\mathsf{tr}(q)).\mathsf{tr}(r) = \mathsf{tr}(\mathsf{succ}(q)).\mathsf{tr}(r) = \mathsf{tr}(\mathsf{succ}(\underline{q}.r))$.

Now let $q.\underline{r}$. Since the term q is headless, r has urgency $\mathsf{urg}(r) > \mathsf{urg}(q) \geq 1$. Let $r = s.\underline{x}.t$ for some action x. If we are strict, we also need to handle the cases where r equals to $s.\underline{x}$, $\underline{x}.t$, and $\underline{x}$. We omit them to avoid repetition. Since $\mathsf{urg}(x) = \mathsf{urg}(r) > 1$, the term x can only be a choice $\bigcirc^{\mathsf{u}} P$ or a non-terminal A. Both of these terms are headless, so we apply the induction hypothesis and obtain $\mathsf{succ}(\mathsf{tr}(x)) = \mathsf{tr}(\mathsf{succ}(x))$. Because $\mathsf{urg}(x) > \mathsf{urg}(q.s)$, the leading subterm is $\mathsf{tr}(q).\mathsf{tr}(s).\underline{\mathsf{tr}(x)}.\mathsf{tr}(t)$. Indeed, $\mathsf{urg}(\mathsf{tr}(q.s)) \leq \max(1, \mathsf{urg}(q.s))$ and $\overline{\mathsf{urg}(\mathsf{tr}}(q.s)) < \mathsf{urg}(\mathsf{tr}(x))$. Finally, we derive

$$\begin{aligned}
\mathsf{succ}(\mathsf{tr}(q.\underline{r})) &= \mathsf{succ}(\mathsf{tr}(q.s.\underline{x}.t)) \\
&= \mathsf{tr}(q.s).\mathsf{succ}(\mathsf{tr}(x)).\mathsf{tr}(t) \\
&= \mathsf{tr}(q.s).\mathsf{tr}(\mathsf{succ}(x)).\mathsf{tr}(t) \\
&= \mathsf{tr}(\mathsf{succ}(q.s.\underline{x}.t)) \\
&= \mathsf{tr}(\mathsf{succ}(q.\underline{r}))\ \blacksquare
\end{aligned}$$

Proof of Lemma 24: We show both directions by an induction on the number of moves Eve needs to win.

Forward Direction: Let $p \in \mathbf{T}$ and $r \in \mathsf{QRun}_{\mathsf{hd}(p)}$. For the base case, let Eve wins O from p in 0 moves. Then, $\mathsf{hd}(p) = p \in \mathbf{W}$ and $\mathsf{tl}(p) = \mathsf{skip}$. Because Eve wins, p can not contain err. Then, $r \in \mathsf{QRun}_p$ must be of the form $(i, p_1).(p_1, p_2)\dots(p_{n-1}, p_n).\mathsf{tr}(a_n)\dots\mathsf{tr}(a_{k-1})$. We know that the word $w = a_0 \dots a_n \dots a_{k-1}$ that corresponds to $p \overset{*}{=} w$ has a run on the DFA for O. Per definition of QRun_p, the DFA runs w from i to p_n in $n-1$ steps, and Eve can choose the remaining transitions to reach $p_k \in \mathcal{F}$.

For the inductive case, let Eve reach O from p in β moves. Per Lemma 25, this also holds from $w.q$, where $w = \mathsf{hd}(p)$ and $q = \mathsf{tl}(p)$. We show that $r \Downarrow \mathsf{tr}(O)$ for some $r_w \in \mathsf{QRun}_w$ and $r \overset{*}{=} r_w.\mathsf{tr}(q)$. Per Lemma 25, showing this for one such r suffices. We first observe that Eve has a strategy to reach

$r'_w.\underline{\mathsf{tr}(q)}$ where $r'_w \in \mathsf{QRun}_w$. In case of $\mathsf{urg}(q) \geq 2$ we have $r_w = r'_w$ and the leading subterm $r_w.\underline{\mathsf{tr}(q)}$ due to $\mathsf{urg}(r_w) \leq 1 < \mathsf{urg}(\mathsf{tr}(q))$. Otherwise, $\mathsf{urg}(\mathsf{tr}(q)) = \mathsf{urg}(q) = 1$ (it can't be 0 because $q = \mathsf{tl}(p)$). If $r_w \in \mathsf{QRun}_w \cap \mathbf{W}$, we already have $r_w.\underline{\mathsf{tr}(q)}$. So let $r_w \in \mathsf{QRun}_w \setminus \mathbf{W}$, i.e. $\mathsf{urg}(r_w) = 1$. Eve resolves each term of the form $\bigvee^1\{(q,r) \mid r \in \delta(q,a)\}$ for some $a \in \Sigma$ and extend the determined part of the run in QRun_w. We know that $w \stackrel{*}{\neq} \mathsf{err}$, because Eve wins $w.q$. So, per definition of QRun_w, Eve can find fitting transitions. Exhaustive application of this strategy rewrites r_w to $r'_w \in \mathsf{QRun}_w \cap \mathbf{W}$ for the desired $r'_w.\underline{\mathsf{tr}(q)}$.

We now show $r'_w.\mathsf{tr}(q) \Downarrow \overline{\mathsf{tr}(O)}$. Let $\mathsf{own}(w.q) = Eve$. The case $\mathsf{own}(w.q) = Adam$ is dual. Remember $p = w.q$ and $w = \mathsf{hd}(p)$, so we have $w.\underline{q}$. Then, there is a $w.q' \in \mathsf{succ}(w.q)$ from which Eve wins in $\gamma < \beta$ moves. Since $w.q' \stackrel{*}{=} w.\mathsf{hd}(q').\mathsf{tl}(q')$, Lemma 25 tells us that Eve also wins from $w.\mathsf{hd}(q').\mathsf{tl}(q')$. We have $r'_w.\mathsf{tr}(\mathsf{hd}(q')) \in \mathsf{QRun}_{w.\mathsf{hd}(q')} = \mathsf{QRun}_{\mathsf{hd}(w.q')}$, so we can apply the induction hypothesis to see that Eve wins from $r'_w.\mathsf{tr}(\mathsf{hd}(q')).\mathsf{tr}(\mathsf{tl}(q'))$ in γ turns. The fact $r'_w.\mathsf{tr}(\mathsf{hd}(q')).\mathsf{tr}(\mathsf{tl}(q')) \stackrel{*}{=} r'_w.\mathsf{tr}(q')$ and Lemma 25 imply that $r'_w.\mathsf{tr}(q') \Downarrow \mathsf{tr}(O)$. Since q is headless, Lemma 26 tells us $\mathsf{succ}(\mathsf{tr}(p)) = \mathsf{tr}(\mathsf{succ}(p))$ and thus $r'_w.\mathsf{tr}(q') \in \mathsf{succ}(r'_w.\underline{\mathsf{tr}(q)})$. Then, we also have $r'_w.\mathsf{tr}(q) \Downarrow \mathsf{tr}(O)$. This concludes this direction of the proof.

Backward Direction: Let $p \in \mathbf{T}$, $w = \mathsf{hd}(p)$, $q = \mathsf{tl}(p)$ and $O \subseteq \Sigma^*$. For the base case, let Eve win $r_w.\mathsf{tr}(q)$ in 0 moves. Then, $\mathsf{tr}(q)$ is a command and this is only possible if $q = \mathsf{skip}$. Since $r_w \in \mathsf{tr}(O)$, the run r_w is accepting in the DFA for O. So, $w = p \in O$.

For the inductive case, let Eve win $r_w.\mathsf{tr}(q)$ in β moves for some $r_w \in \mathsf{QRun}_w$. The first case is $\underline{r_w}.\mathsf{tr}(q)$. Then, there is a term $\mathsf{tr}(a)$ in r_w, so Eve can simply choose the corresponding transition in the DFA to extend r_w to $r'_w \in \mathsf{QRun}_w$ per definition of QRun_w. Eve then wins from $r'_w.\mathsf{tr}(q)$ in $\gamma < \beta$ moves. We can apply the induction hypothesis to see that Eve wins from $w.q$, and per Lemma 25 from p. The second case is $r_w.\underline{\mathsf{tr}(q)}$. Let $\mathsf{own}(r_w.\mathsf{tr}(q)) = Eve$. Again, the case $\mathsf{own}(r_w.\overline{\mathsf{tr}(q)}) = Adam$ is dual. Since q is headless, we have $\mathsf{succ}(\mathsf{tr}(q)) = \mathsf{tr}(\mathsf{succ}(q))$ per Lemma 26. Then, Eve wins from some $r_w.\mathsf{tr}(q')$ where $q' \in \mathsf{succ}(q)$ in $\gamma < \beta$ moves. Write $r_w.\mathsf{tr}(q') \stackrel{*}{=} r_w.\mathsf{tr}(\mathsf{hd}(q')).\mathsf{tr}(\mathsf{tl}(q'))$. As in the previous direction, we have $r_w.\mathsf{tr}(\mathsf{hd}(q')) \in \mathsf{QRun}_{w.\mathsf{hd}(q')}$. Per definition, $\mathsf{tl}(q')$ is headless. We can apply the induction hypothesis to get that Eve wins from $w.\mathsf{hd}(q').\mathsf{tl}(q') \stackrel{*}{=} w.q' \in \mathsf{succ}(w.\underline{q})$. So Eve wins from $w.q \stackrel{*}{=} p$ as well, completing the proof. ■

Proof of Lemma 15 for $\mathsf{h} = 1$:

Preliminary Facts: Before moving on to the computation of $\chi(\mathbf{C}_{\mathsf{tr}(O)})$ for $\mathsf{h} = 1$, we study $SNF_0^{\mathsf{tr}(O)}$ more closely.

Lemma 27: The syntactic monoid of $\mathsf{tr}(O)$ is $SNF_0^{\mathsf{tr}(O)} = Q^2 \cup \{\mathsf{err}, \mathsf{skip}\}$. Furthermore, the syntactic equivalences $(p,q).(q,r) \equiv_{\mathsf{tr}(O)} (p,r)$ and $(p,q).(s,r) \equiv_{\mathsf{tr}(O)} \mathsf{err}$ hold for all $p, q, r, s \in Q$ with $q \neq s$.

Proof: First we show that $Q^2 \cup \{\mathsf{err}, \mathsf{skip}\} \subseteq SNF_0^{\mathsf{tr}(O)}$, i.e. that these elements are pairwise not equivalent. It is clear to see that for any $(p,q) \in Q^2$, $(i,p).(p,q).(q,f) \Downarrow O$, so we get $(p,q) \not\equiv_{\mathsf{tr}(O)} \mathsf{err}$ and thus $(p,q) \not\sqsubseteq_{\mathsf{tr}(O)} \mathsf{err}$ per soundness. It is also clear to see that for any $(p,q) \neq (s,r) \in Q^2$, we have $(i,p).(p,q).(q,f) \Downarrow \mathsf{tr}(O)$ but $(i,p).(s,r).(q,f) \Downarrow \mathsf{tr}(O)$ fails to hold since $p \neq s$ or $q \neq r$. skip is the neutral element of the syntactic monoid and unless $|Q| = 1$ it is different from any $(p,q) \in Q^2$.

Now, we show $SNF_0^{\mathsf{tr}(O)} \subseteq Q^2 \cup \{\mathsf{err}, \mathsf{skip}\}$. These are exactly the terminal symbols, so it will suffice to show that the set $Q^2 \cup \{\mathsf{err}\}$ is closed under concatenation up to $\equiv_{\mathsf{tr}(O)}$. Concatenations that involve err or skip elements are reduced by Axiom (M) to an element from $Q^2 \cup \{\mathsf{err}, \mathsf{skip}\}$.

It remains to show the claimed congruences. For that, let $p, q, r, s \in Q$ with $q \neq s$ and let $c[\bullet] = s.\bullet.t$ be a concatenative context with $s, t \in \mathbf{W}$. The first observation is that $c[(p,q).(r,s)]$ is losing for Eve, since she can not form a continous run, i.e. she can not reach $\mathsf{tr}(O)$. Then, per Axiom (S) we get $(p,q).(s,r) \equiv_{\mathsf{tr}(O)} \mathsf{err}$. An accepting, continous run needs to have only three properties. Namely, it must start from i, it must not have discontinuities, and it must end at some $f \in \mathcal{F}$. But, $(p,q).(q,r)$ and (p,r) are both continuous while starting and ending at the same states. So we see that $c[(p,q).(q,r)]$ is an accepting run if and only if $c[(p,r)]$ is an accepting run. Thus, $(p,q).(q,r) \equiv_{\mathsf{tr}(O)} (p,r)$. ■

We define $\delta[w] = \{(p,q) \in Q^2 \mid p \xrightarrow{v} q, w \stackrel{*}{=} v \in \Sigma^+\}$ for all $w \in \mathbf{W}$ without err that are not skip.

Lemma 28: $\mathsf{tr}(w) \equiv_{\mathsf{tr}(O)} \bigvee^1 \delta[w]$.

Proof: This is done by induction on $|w|$, the number of symbols contained in w. For the base case, we have $|w| = 1$. Then, $\mathsf{tr}(w) = \bigvee^1\{(p,q) \mid q \in \delta(p,w)\}$. For the inductive case, let $w.v \in \mathbf{W}$ unequal to err or skip by $\stackrel{*}{=}$. Then,

$$\begin{aligned}\mathsf{tr}(w.v) &= \mathsf{tr}(w).\mathsf{tr}(v)\\ &\equiv_{\mathsf{tr}(O)} \bigvee^1 \delta[w].\bigvee^1 \delta[v]\end{aligned}$$

Applying (D1) and (D2) and flattening with (L4):

$$\equiv_{\mathsf{tr}(O)} \bigvee^1\{(p,q).(s,r) \mid (p,q) \in \delta[w], (s,r) \in \delta[v]\}$$

Using Lemma 27 and (L5) to remove inconsistent runs:

$$\begin{aligned}&\equiv_{\mathsf{tr}(O)} \{(p,q).(q,s) \mid (p,q) \in \delta[w], (q,r) \in \delta[v]\}\\ &\equiv_{\mathsf{tr}(O)} \bigvee^1 \delta[w.v]\end{aligned}$$

■

1) Iterating the Characteristic Terms: Recall that we have constructed $TNF_1^{\mathsf{tr}(O)}$ to be the image of SNF_1^O under $\mathsf{nf}_{\mathsf{tr}(O)} \circ \mathsf{tr}$. This poses a problem for $\mathsf{h} = 1$. An Adam choice alone can not represent the image of a $SANF_1^O$ term under normalization. Namely, for some $\bigvee_{i\in I}^1 \bigwedge^1 P_i \in SNF_1^{\mathsf{tr}(O)}$, we might not have any $p \in \mathbf{T}$ with $\mathsf{tr}(p) \equiv_{\mathsf{tr}(O)} \bigwedge P_i$. So, if we naively apply the construction for $\mathsf{h} = 1$, we are forced to iterate over a substantial subset of $SNF_1^{\mathsf{tr}(O)}$. This results in a

$\exp_2(\mathcal{O}(|O|^2))$ time complexity. For this reason, we compute $\chi(\mathbf{C}_{\mathsf{tr}(O)})$ directly, by exploiting the Myhill-Nerode right-precongruence on the states of the DFA.

Definition 7: For any $p, q \in Q$, we have $p \leq_N q$ if and only if for all $w \in \Sigma^*$, $p \xrightarrow{w} f \in \mathcal{F}$ implies $q \xrightarrow{w} f'$ for some $f' \in \mathcal{F}$.

We extend this to pairs of states $(p, p'), (q, q') \in Q^2$. We let $(p, p') \leq_N (q, q')$ if and only if $p = q$ and $p' \leq_N q'$. We call a state pair (p, q) dead, if there is no run from q to a state in $\mathcal{F}$ in the DFA for O. For some $c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}$, we call $\mathcal{S}(c[\bullet]) = \{p \in SNF_0^{\mathsf{tr}(O)} \mid c[p] \Downarrow O\}$ the solution space of $c[\bullet]$. Towards computing $\chi(\mathbf{C}_{\mathsf{tr}(O)})$, we show two important facts that expose the relationship between the extended $\leq_N$ and solution spaces of contexts.

First, we show that there is a context $c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}$ for each not-dead $(p, q) \in Q^2$, where $\mathcal{S}(c[\bullet])$ is the $\leq_N$-upward closure of (p, q).

Lemma 29: For all $(p, q) \in Q^2$ that is not dead, there is a $c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}$ with $\mathcal{S}(c[\bullet]) = \{(p', q') \in Q^2 \mid (p, q) \leq_N (p', q')\}$.

Proof: We claim that $c[\bullet] = (q_0, p).\bullet.\bigwedge^1_{w \in P} \mathsf{tr}(w)$ where $P = \{v \mid q \xrightarrow{v} q_f \in \mathcal{F}\}$. Note that $P \neq \emptyset$, because (p, q) is not dead. For $(s, t) \in Q^2$ with $p \neq r$, $c[(s, t)]$ is losing for Eve, since $(q_0, p).(s, t) \equiv_O \mathsf{err}$ per Lemma 27. Now assume $(p, s) \in Q^2$. We use Lemma 28 and get $c[(p, s)] \equiv_{\mathsf{tr}(O)} (q_0, p).(p, s).\bigwedge^1_{w \in P} \bigvee^1 \mathsf{Rch}_w$. Then, Eve wins $c[(p, s)]$ if and only if Eve wins $(q_0, p).(p, s).\bigvee^1 \mathsf{Rch}_w$ for all $w \in P$. This is equivalent to w having a run from s to some $q_f \in \mathcal{F}$ for all $w \in P$, This is the definition of $q \leq_N s$. ∎

Then, we show that for all $c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}$, the solution spaces are upward closed.

Lemma 30: Let $c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}$ and $(p, q), (p', q') \in Q$ with $(p, q) \leq_N (p', q')$. Then, $(p, q) \in \mathcal{S}(c[\bullet])$ implies $(p', q') \in \mathcal{S}(c[\bullet])$.

Proof of Lemma 30: Wlog. let $(p, q), (p, q') \in Q^2$ with $(p, q) \leq_N (p, q')$ and $c[\bullet] = (s, t).\bullet.\bigvee^1_{i \in I} \bigwedge^1_{w \in P_i} \bigvee^1 \mathsf{Rch}_w$. If $s \neq q_0$ or $t \neq p$ no matter which (p, r) is inserted to this context, Eve can never derive a continious run. Now let $c[\bullet] = (q_0, p).\bullet.\bigvee^1_{i \in I} \bigwedge^1_{w \in P_i} \bigvee^1 \mathsf{Rch}_w$ and assume that Eve wins from $c[(p, q)]$. Then, there is an $i \in I$ where for all $w \in P$, Eve wins $(q_0, p).(p, q).\bigvee^1 \mathsf{Rch}_w$. So, there is an $i \in I$ where for all $w \in P_i$, there is a run $q \xrightarrow{w} q_f$ for some $q_f \in \mathcal{F}$. The latter part of the statement and $(p, q) \leq_N (p, q')$ implies that there also is a run $q' \xrightarrow{w} q'_f$ for some $q'_f \in \mathcal{F}$. So, from $c[(p, q')]$, Eve plays the same $i \in I$. Let Adam play some $w \in P_i$. The resulting term is $(q_0, p).(p, q').\bigvee^1 \mathsf{Rch}_w$. So Eve chooses $(q', f') \in \mathsf{Rch}_w$ to reach $(q_0, p).(p, q').(q', f')$ and win. ∎

By applying Equation (3) along with Lemma 29 and Lemma 30, we see that a set $X \subseteq SNF_0^{\mathsf{tr}(O)}$ is a solution space if and only if it is an upward closure that does not contain dead pairs. We can iterate through all left-sides $p \in Q$ and right-sides $Q' \subseteq Q$ and build the upward closures, therefore solution spaces, in $\exp_1(\mathcal{O}(|O|^2))$ time. Recall that for $c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}$, we had $\chi(c[\bullet]) = \bigwedge^1 \mathcal{S}(c[\bullet])$ (Equation (2)). So, we can build $\{\bigwedge^{\mathsf{u}} \mathcal{S}(c[\bullet]) \mid c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}\} = \{\chi(c[\bullet]) \mid c[\bullet] \in \mathbf{C}_{\mathsf{tr}(O)}\} = \chi(\mathbf{C}_{\mathsf{tr}(O)})$ in $\exp_1(\mathcal{O}(|O|^2))$ time.

J. Hyperproperties and Urgency

Hyperproperties emerged as a unifying approach to information flow and security properties, which cannot be stated as classical safety or liveness properties on a single trace. A hyperproperty relates traces and is formulated over a set of traces rather than a single trace. A novel approach are logics like HyperLTL [4] to describe hyperproperties. We will stay more general and define an n-trace hyperproperty to be any DFA $\mathcal{A}$ over $(\Sigma^n)^*$. Note that we consider finite traces rather than infinite ones. In particular, a hyperproperty talkes about the relation of traces and does so in a highly synchronized manner: A set of traces is accepted by $\mathcal{A}$ ensures that they all share the exact same length. To model different traces of the same system so that the resulting observations are synchronized is a common problem with hyperproperties. We will not tackle this issue here, but assume that the traces of a system $\mathcal{K}$ are partitioned into sets of same length. That means that the set of traces $\mathsf{tr}(\mathcal{K}) \subseteq \Sigma^*$ is partitioned into sets of traces $\mathsf{tr}_l(\mathcal{K}) \subseteq \Sigma^l$ of length $l \in \mathbb{N}$.

Definition 8: Let $\mathcal{K}$ be a system (Kripke structure) with trace-set $\mathsf{tr}(\mathcal{K}) \subseteq \Sigma^*$. An n-trace hyperproperty $\mathcal{A}$ is satisfied by $\mathcal{K}$, $\mathcal{K} \models \mathcal{A}$, if there is $l \in \mathbb{N}$ such that

$$\exists w_1 \in \mathsf{tr}_l(\mathcal{K}) \forall w_2 \in \mathsf{tr}_l(\mathcal{K}) \dots Q w_n \in \mathsf{tr}_l(\mathcal{K}). \prod_{i=1}^{n} w_i \in \mathcal{L}(\mathcal{A})$$

The definition might be non-intuitive at first glance: Only for a single length l, the set $\mathsf{tr}_l(\mathcal{K})$ has to satisfy the hyperproperty. Further, every hyperproperty begins to quantify with an $\exists$ quantifier. This makes it impossible to formulate hyperproperties from the $\forall\exists$-fragment of HyperLTL. However, the ability to deterministically decide hyperproperties of the above shape is already sufficient to decide any hyperproperty, including ones that start with $\forall$, for which we check its negation.

We show how to model check a hyperproperty by translating the system $\mathcal{K}$ and the hyperproperty $\mathcal{A}$ into a term p and an objective O, such that $\mathcal{A}$ is satisfied by $\mathcal{K}$ if and only if $p \Downarrow O$. Intuitively, we model the quantifiers in Definition 8 by the players. Each quantifier is resolved in one level of urgency throughout the whole term. Only then, and in knowledge of the resolution of previous quantifiers, a player resolves the next quantifier. Formally, we define the terms p_i for $i \in [1, n+1]$ over the alphabet $\Sigma = \delta$ by induction:

$$p_{n+1} = \mathsf{skip} \qquad p_i = \begin{cases} \bigvee^{n-i+1} \delta.p_{i+1} & i \text{ odd} \\ \bigwedge^{n-i+1} \delta.p_{i+1} & i \text{ even} \end{cases}$$

The final term p is a single non-terminal $p = A$ with $E(A) = \mathsf{skip} \vee^n p_1.A$. This lets Eve choose the first trace and while doing so, she also fixed the length of the considered traces.

For the objective O, we want to capture the hyperproperty $\mathcal{A}$, which already is a DFA. For the transformation to O we only need to transform the input alphabet from δ^n to δ and

make sure the players choose actual traces of the system $\mathcal{K}$. Technically, for a word $w \in (\Sigma^n)^*$, we obtain $w_i \in \Sigma^*$ by restricting w to the ith component. We define the flattening flat of a word in $(\Sigma^n)^*$ to Σ^* inductively by $\mathsf{flat}(\mathsf{skip}) = \mathsf{skip}$ and $\mathsf{flat}((t_1, \dots, t_n).v) = t_1 \dots t_n.\mathsf{flat}(v)$. We set the set of correct runs on component i by $O_i = \{w \in (\Sigma^n)^* \mid w_i \in \mathsf{tr}(\mathcal{K})\}$.

$$O = \mathsf{flat}(\mathcal{L}(\mathcal{A}) \cap \bigcap_{i \text{ odd}} O_i \cup \bigcup_{i \text{ even}} \overline{O_i})$$

Note that $|O|$ is linear in $|\mathcal{A}| + |\mathcal{K}|$.

Theorem 6: $\mathcal{K} \models \mathcal{A}$ if and only if $p \Downarrow O$.

Proof: By construction, $p \Downarrow O$ if and only if there is $l \in \mathbb{N}$ with $(p_1)^l \Downarrow O$, because unrolling A an infinite number of times yields the win to Adam.

Next we inspect the terms $p(w_1, \dots, w_m)$ that can occur whenever urgency $n - m$ is next to be resolved. Here, w_i are from $\mathsf{tr}_l(\mathcal{K})$ and for odd i are chosen corresponding to the $\exists$-quantifiers. By construction, $p(w_1, \dots, w_m)$ has shape

$$t_{1,1} t_{2,1} \dots t_{m,1} p_{m+1} \ \dots \ t_{1,l} t_{2,l} \dots t_{m,l} p_{m+1} \,.$$

where $w_i = t_{i,1} \dots t_{i,l}$.

We prove the statement: Eve wins $p(w_1, \dots, w_m) \Downarrow O$ if and only if m is even and there is $w_{m+1} \in \mathsf{tr}_l(\mathcal{K})$ with $p(w_1, \dots, w_m, w_{m+1}) \Downarrow O$, or m is odd and for all $w_{m+1} \in \mathsf{tr}_l(\mathcal{K})$ holds $p(w_1, \dots, w_m, w_{m+1}) \Downarrow O$.

If m is even, then Eve can choose a sequence $w_{m+1} \in \delta^l$ to transform $p(w_1, \dots, w_m)$ into $p(w_1, \dots, w_m, w_{m+1})$. If she chooses a word outside $\mathsf{tr}_l(\mathcal{K})$ she loses due to O_{m+1}. Thus, she wins if and only if there is $w_{m+1} \in \mathsf{tr}_l(\mathcal{K})$ she can choose and $p(w_1, \dots, w_m, w_{m+1}) \Downarrow O$.

The case of m odd is similar.

Finally, $n = m$ makes $p_{m+1} = \mathsf{skip}$, so $p(w_1, \dots, w_n) \Downarrow O$ if and only if $p(w_1, \dots, w_n) \in \mathsf{flat}(\mathcal{L}(\mathcal{A}))$ by definition. ■

The complexity of checking an n-trace hyperproperty $\mathcal{K} \models \mathcal{A}$ using this approach is $(2n-1)-\mathsf{EXPTIME}$ bounded (Corollary 1). This is still far from optimal, considering that n-trace hyperproperties can be decided in $(n-1)-\mathsf{EXPSPACE}$ [4]. The overhead stems from the shape of normalforms, which allow for Adam and Eve choices in each layer of urgencies, while our approach produces only one type of nondeterminism for each urgency. We intend to investigate this special case in a separate study.

K. Model Checking Hyperproperties for Recursive Programs

Checking hyperproperties on pushdown systems is known to be undecidable in the general case [5]. But showing it undecidable does not satisfy the desire to check recursive programs against hyperproperties. Indeed, a first approach was proposed directly in [5], where one type of quantifier has to work with finite state approximations instead of the actual system. We take a different route to restrict the general setting. To motivate the restriction, consider the algorithm in Listing 2 for multiplication of high-bit numbers.

```
bit[] KM(bit[] x, bit[] y) {
    if (len(x) < 64) return (int) x * (int) y;

    mid = len(x) / 2;
    x1 = x[:mid];
    y1 = y[:mid];
    x2 = x[mid+1:];
    y2 = y[mid+1:];

    z2 = KM(x1, y1);
    z0 = KM(x2, y2);
    z1 = KM(x1 + x2, y1 + y2) - z2 - z0;

    return (z2 << 2mid) + (z1 << mid) + z0;
}
```

Listing 2: Karatsuba multiplication.

The algorithm performs a high-bit multiplication of two numbers `x` and `y`. Instead of naively multiplying 64-bit windows of the bit streams, the algorithm recursively splits the inputs in half. It performs 3 multiplications on the half-sized integers and adds them together for the final result. The actual algorithm needs some more care for bit-overflows, but we shall ignore it here.

Algorithms like Listing 2 are used by security protocols like `OpenSSL` [40]. A known weakness of security protocols are timing-based attacks [41], [42]. In these attacks, one does not run the program once to derive a secret. Instead, we measure its execution time over multiple runs with different controllable inputs and deduct secret values used by the program from the measured execution times. So, safety from timing attacks can be obtained by requiring a 2-trace hyperproperty on `KM`: For all traces w, v of `KM`, their execution time does not differ. We do not go into modelling details of how to phrase this property as a DFA $\mathcal{A}$. But we make one crucial discovery when we want to compare multiple runs of the Karatsuba algorithm: Its recursion depth is only dependent on the length of `x` and `y`, a parameter that is very often public knowledge. Even more important, runs of different recursion depth have close to no chance for a similar execution time. The previously stated hyperproperty would very likely not be satisfied. But when the recursion depth parameter is usually known, we instead want to ask for a different hyperproperty to hold: For all recursion depths d, and for all traces w, v with recursion depth d, their execution time does not differ. More generally speaking, the traces w and v agree on their *recursive structure*. We utilize this observation and restrict our check for a hyperproperty $\mathcal{A}$ to sets of traces that agree on their recursive structure.

1) Recursive Programs: We consider the following simple language, where we abstract away from the actual commands and focus on the recursion principle of the language.

$$\begin{aligned} c &::= a \mid \texttt{f()} \mid c.c \mid \texttt{if } e \texttt{ } c\texttt{:}c \\ \texttt{f} &::= \texttt{f() -> } c\texttt{.return} \end{aligned}$$

We have commands $a \in \Sigma$ to modify the state, function calls `f()` and concatenation. Parameters and return values are passed to/from `f` by state-manipulation. A program $\mathcal{P} = (\mathcal{F}, E)$ is a set of function symbols $\mathcal{F}$ with a distinguished initial function $\texttt{main} \in \mathcal{F}$ and the function definitions E.

2) Semantics and Recursion Structure: We assume the domain of booleans $\mathcal{D} = [\mathcal{V} \to \mathbb{B}]$. The semantics are kept

arbitrary for expressions $[\![e]\!] : \mathcal{D} \to \mathbb{B}$ and commands $[\![a]\!] : \mathcal{D} \to \mathcal{D}$. Semantics of function calls and concatenation are as expected. `if`-statements branch left on non-zero evaluation and right on zero. For the further development, we also require a special command `assume` $e \in \Sigma$. Its semantics operate on a fresh variable η and are tuned to observe whether an `assume` command failed.

$$[\![\texttt{assume}\ \ e]\!](d) = \begin{cases} d & [\![e]\!](d) = \texttt{true} \\ d[\eta \mapsto \texttt{true}] & [\![e]\!](d) = \texttt{false} \end{cases}$$

A *branching* of $\mathcal{P}$ is a finite tree t (a prefix closed subset of $\mathbb{N}^*$) with a labelling cmd. The labelling assigns nodes of t commands a, function calls `f`, and `return` statements. The root node is labeled with $\mathsf{cmd}(\varepsilon) = \texttt{main}$. Nodes tn labelled by $\mathsf{cmd}(tn) = a$ or `return` are leaves. A $\mathsf{cmd}(tn) = \texttt{f}(n)$-labelled node has children corresponding to one branch of their body c. The set of branches $\mathsf{br}(c)$ is

$$\mathsf{br}(a) = \{a\} \quad \mathsf{br}(\texttt{f}()) = \{\texttt{f}\} \quad \mathsf{br}(c_1.c_2) = \mathsf{br}(c_1).\mathsf{br}(c_2)$$
$$\mathsf{br}(\texttt{if}\ \ e\ \ c_1 \texttt{:} c_2) = \texttt{assume}\ \ e.\mathsf{br}(c_1) \cup \texttt{assume}\ \ !e.\mathsf{br}(c_2)$$

So if $w \in \mathsf{br}(c)$ is the chosen branch, then tn has $|w|$ children labelled by the corresponding a, `f`, or `return` in w.

A *trace* of $\mathcal{P}$ is a pair $(\mathsf{cmd}, \mathsf{st})$ where cmd is a branching with domain t and $\mathsf{st} : t \to \mathcal{D}$ is a state assignment. Every node tn is labelled by $\mathsf{st}(tn) \in \mathcal{D}$, the state of the program before execution of $\mathsf{cmd}(tn)$, $\mathsf{st}(\varepsilon) \in \mathcal{D}$ is the input state. Consider node tn with $\mathsf{cmd}(tn) = \texttt{f}$, so the children are one branch of its defining body. Its leftmost child $tn.0$ inherits the state, $\mathsf{st}(tn.0) = \mathsf{st}(tn)$. The rightmost child $tn.m$ has label $\mathsf{cmd}(tn.m) = \texttt{return}$ and yields back $[\![\texttt{f}]\!](\mathsf{st}(tn)) = \mathsf{st}(tn.m)$. Intermediate children $tn.i$ just carry out their semantics to the next node by $\mathsf{st}(tn.i+1) = [\![\mathsf{cmd}(tn.i)]\!](\mathsf{st}(tn.i))$.

A *recursion structure* r for $\mathcal{P}$ is a tree labelling $r : t \to \mathcal{F}$. The internal of a tree t is the set $\mathsf{int}(t) = \{tn \mid tn.0 \in t\}$.

Definition 9: A branching cmd has recursion structure r if $\mathsf{int}(\mathrm{dom}\,\mathsf{cmd}) = \mathrm{dom}\, r$ and cmd and r coincide on $\mathrm{dom}\, r$.

The recursive structure thus identifies runs that differ only in st and the actual commands executed, but the function calls are exactly the same. We say $(\mathsf{cmd}, \mathsf{st})$ is an r-trace when cmd has recursion structure r. A recursion structure r is proper, if there is an r-trace of $\mathcal{P}$.

3) Checking Hyperproperties for similar Recursion Structure: We define the yield $\mathrm{yld}(t)$ of a tree as usual. The set of trace-observations with recursion structure r is the set $\mathsf{tr}_r(\mathcal{P}) = \{\mathrm{yld}(\mathsf{cmd}) \mid (\mathsf{cmd}, \mathsf{st}) \text{ is an } r\text{-trace of } \mathcal{P}\} \subseteq \Sigma^*$.

Definition 10: Let $\mathcal{A}$ be an n-trace hyperproperty. A program $\mathcal{P}$ satisfies $\mathcal{A}$, $\mathcal{P} \models \mathcal{A}$, if there is a proper recursion structure r with

$$\exists w_1 \in \mathsf{tr}_r(\mathcal{P}) \forall w_2 \in \mathsf{tr}_r(\mathcal{P}) \ldots Q w_n \in \mathsf{tr}_r(\mathcal{P}). \prod_{i=1}^{n} w_i \in \mathcal{L}(\mathcal{A})$$

To present our approach, we assume that programs are prefix-branching. We call a program $\mathcal{P}$ *prefix-branching* when function calls are never succeeded by commands $a \in \Sigma$. Formally, we assume the program code stems from the following, slightly different grammar:

$$\begin{aligned} f &::= \texttt{f}() \mid f.f \\ c &::= a \mid c.c \\ pb\text{-}c &::= f \mid c.pb\text{-}c \mid \texttt{if}\ e\ pb\text{-}c\texttt{:}pb\text{-}c \\ \texttt{f} &::= \texttt{f}()\ \texttt{->}\ pb\text{-}c.\texttt{return} \end{aligned}$$

Note, that this is not only a presentational decision. While every program $\mathcal{P}$ has an equivalent representation in prefix-branching form, the translation incurs more functions and thus, fixing a recursion structure may fix more behavior (e.g. branching beyond function calls) than desired. This means that we actually restrict our class of programs by enforcing prefix-branching. We do not exactly know, how impactful the restriction is for practical code, but our example (Listing 2) exerts the desired structure (up to parameter passing).

Theorem 7: $\mathcal{P} \models \mathcal{A}$ is decidable for prefix-branching $\mathcal{P}$.

Similar as for the finite state case, we will translate $\mathcal{P}$ and $\mathcal{A}$ into a term p and an objective O. The intent is to first fix a recursion structure, and then replay the call of a function `f` with function body c by the rewriting of a non-terminal `f` into a term $p(\texttt{f})$. Term $p(\texttt{f})$ basically chooses the branch through c. For prefix-branching, a branch is a word from $\Sigma^*\mathcal{F}^*$. With a fixed recursive structure also the sequence of function calls in c is fixed. For a fixed sequence of function calls $f = \texttt{f}_1 \ldots \texttt{f}_m$ the set of available prefixes is captured by $\mathsf{br}_f(c) \subseteq \Sigma^*$ with

$$\mathsf{br}_f(c).f = \mathsf{br}(c) \cap \Sigma^*.f\,.$$

We assume that all command prefixes (the Σ^* part) in $\mathsf{br}_f(c)$ share the same length, $\mathsf{br}_f(c) \subseteq \Sigma^l$ for some $l \in \mathbb{N}$. Similar to the finite state case this is a modelling issue for synchronization towards $\mathcal{A}$. It can be achieved by introducing skips to branches too short. Since $\mathsf{br}(c)$ is finite, there is only a finite set of occurring function call sequences $f(c) = \{f \mid \mathsf{br}_f(c) \neq \emptyset\}$. Finally, we define $p(\texttt{f})$ in a term game $(\mathcal{F}, E)$ with $E(\texttt{f}) = p(\texttt{f})$.

$$p_{n+1}^{c,f}(w_1, \ldots, w_n) = \mathsf{flat}(w_1, \ldots, w_n)$$
$$p_i^{c,f}(w_1, \ldots, w_{i-1}) = \begin{cases} \bigvee_{w_i \in \mathsf{br}_f(c)}^{n-i+1} p_{i+1}^{c,f}(w_1, \ldots, w_i) & i \text{ odd} \\ \bigwedge_{w_i \in \mathsf{br}_f(c)}^{n-i+1} p_{i+1}^{c,f}(w_1, \ldots, w_i) & i \text{ even} \end{cases}$$
$$p(\texttt{f}) = \bigvee_{f \in f(c)}^{n} p_1^{c,f}().f$$

It remains to model the objective O. As before, we construct sets $O_i \subseteq (\Sigma^n)^*$ for $i \in [1, n]$.

$$O = \mathsf{flat}(\mathcal{L}(\mathcal{A}) \cap \bigcap_{i\mathrm{odd}} O_i \cup \bigcup_{i\mathrm{even}} \overline{O_i})$$
$$O_i = \{w \in (\Sigma^n)^* \mid [\![w_i]\!](\eta) = \texttt{true}\}$$

A command sequence $w \in \Sigma^*$ operates on the specific finite domain $\mathcal{D}$. These language is regular because $\mathcal{D}$ is finite.

Theorem 8: $\texttt{main} \Downarrow O$ if and only if $\mathcal{P} \models \mathcal{A}$.

Proof Sketch: Notice that the set of all reachable word terms from $p_1^{c,f}$ is exactly $\mathsf{flat}(\mathsf{br}_f(c)^n)$. So by construction, playing from `main` until the term has no more non-terminals yields a term p_r for some recursion structure r, where the reachable word terms form precisely the set $\mathsf{flat}(\mathsf{br}_r(\mathcal{P})^n)$, where $\mathsf{br}_r(\mathcal{P}) = \{\mathrm{yld}(\mathsf{cmd}) \mid \mathsf{cmd} \text{ is an } r\text{-branching}\}$. As before, the alternative is that there is an infinite rewriting of non-terminals, in which case Eve loses. Thus, $\texttt{main} \Downarrow O$ if and only if there is r such that $p_r \Downarrow O$.

Next we again inspect the terms $p(w_1, \ldots, w_m)$ that can occur whenever urgency $n - m$ is next to be resolved (at first, we have $p_r = p()$ with $m = 0$). This time, w_i are from $\mathsf{tr}_r(\mathcal{P})$ and for odd i are chosen corresponding to the $\exists$-quantifiers. By construction, $p(w_1, \ldots, w_m)$ has shape

$$p_{m+1}^{c^1,f^1}(w_1^1, \ldots, w_m^1).\ \ldots\ .p_{m+1}^{c^l,f^l}(w_1^l, \ldots, w_m^l)\,,$$

where $c^1 \ldots c^l$ is the depth-first left to right traversal of r.

We prove: $p(w_1, \ldots, w_m) \Downarrow O$ if and only if m is even and there is $w_{m+1} \in \mathsf{tr}_r(\mathcal{P})$ with $p(w_1, \ldots, w_m, w_{m+1}) \Downarrow O$, or m is odd and for all $w_{m+1} \in \mathsf{tr}_r(\mathcal{P})$ we have $p(w_1, \ldots, w_m, w_{m+1}) \Downarrow O$.

If m is even, then Eve can choose a sequence of branches $w_{m+1}^i \in \mathsf{br}_{f^i}(c^i)$ which together form the yield of a branching $w_{m+1} = w_{m+1}^1 \ldots w_{m+1}^l \in \mathsf{br}_r(\mathcal{P})$. She does so by transforming $p(w_1, \ldots, w_m)$ into $p(w_1, \ldots, w_m, w_{m+1})$. If she chooses a branching which has no trace $w_{m+1} \notin \mathsf{tr}_r(\mathcal{P})$ she loses due to O_{m+1}. Thus, she wins if and only if there is $w_{m+1} \in \mathsf{tr}_r(\mathcal{P})$ to choose and $p(w_1, \ldots, w_m, w_{m+1}) \Downarrow O$.

The case of m odd is similar.

The final case of $n = m$ implies that the immediate terms are $p_{m+1}^{c^i,f^i}(w_1^i, \ldots, w_m^i) = \mathsf{flat}(w_1^i, \ldots, w_n^i)$ and by definition of $\Downarrow$ for word terms, $p(w_1, \ldots, w_n) \Downarrow O$ if and only if $p(w_1, \ldots, w_n) \in \mathsf{flat}(\mathcal{L}(\mathcal{A}))$. ∎

L. Lower Bound Details

Construction, Objective: The set of terminal symbols Σ consists of assignments $x \coloneqq val$ and assertions $x =_! val$ of the variables $f_\mathsf{u} \in Q \cup \{-\}$, $s_\mathsf{u} \in Q \cup \{-\}$, and $c_\mathsf{u} \in \{\mathrm{nxt}, \mathrm{sty}\}$ for each $0 < \mathsf{u} \leq \mathsf{h}$, along with a variable $gn \in \{+, -\}$. In the parts of the play, where the urgency of the term is $0 < \mathsf{u} \leq \mathsf{h}$, variable f_u will keep track of the first MPDG state, the variable s_u will keep track of the latest MPDG state, and $c_\mathsf{u} \in \{\mathrm{nxt}, \mathrm{sty}\}$ will be used to enforce the correctness of context switches. The variable $gn \in \{+, -\}$ keeps track of whether the game has generated the second stack by making the first context switch. The objective DFA $\mathcal{A}$ processes the updates and assertions on the values of these variables. For each $0 < \mathsf{u} \leq \mathsf{h}$, the DFA also keeps an assertion failure flag $err_\mathsf{u} \in \{\bot, \top\}$, that records whether there has been assertion failure for f_u, s_u, c_u, and gn. If an assertion failure happens for one of these variables, then err_u is irrevocably set to $\bot$. In the initial state i, we have $gn = f_\mathsf{u} = s_\mathsf{u} = -$, $c_\mathsf{u} = \mathrm{st}$, and $err_\mathsf{u} = \top$ for all $0 < \mathsf{u} \leq \mathsf{h}$. The DFA $\mathcal{A}$ accepts if and only if $s_1 \in F$, there are no assertion errors ($err_\mathsf{u} = \top$ for all $0 < \mathsf{u} \leq \mathsf{h}$), and the latest states are consistent with the first states ($s_{\mathsf{u}+1} = f_\mathsf{u}$ for all $0 < \mathsf{u} < \mathsf{h}$).

Construction, Assignments: We now move on to the construction of the defining assignments. Each stack symbol is represented by a different term for each urgency. The set of non-terminals is $N = \{x_{NT}^\mathsf{u} \mid x \in \Gamma, 0 < \mathsf{u} \leq \mathsf{h}\}$. The representation of an individual stack symbol for urgency u, wraps the corresponding non-terminal in a unary choice with urgency u. Formally the representing term is the singleton choice $x^\mathsf{u} = \bigvee^\mathsf{u} x_{NT}^\mathsf{u}$ for all $0 < \mathsf{u} \leq \mathsf{h}$. This ensures that the u-representation of a stack symbol has urgency u (remember that non-terminals have highest urgency). Furthermore, the term that represent the stack symbol must be the leftmost action. This allows a concatenation of terms that represent stack symbols to act like one stack in the MPDG. The defining assignments $E : N \to \mathbf{T}$ are laid out below for all $x_{NT}^\mathsf{u} \in N$. We use helper terms to simplify the representation. For all $\mathsf{u} \leq \mathsf{h}$, $\mathsf{v} < \mathsf{h}$, $p, q \in Q$, $x \in \Gamma$, and $w \in x^{\leq 2}$ we have:

$$E(x_{NT}^\mathsf{u}) = (c_\mathsf{u} =_! \mathrm{nx}.x^{\mathsf{u}-1}) \vee^\mathsf{u} (c_\mathsf{u} =_! \mathrm{st}.Pop_x^\mathsf{u})$$

$$Pop_x^\mathsf{u} = \bigvee_{p \in Q}^\mathsf{u} s_\mathsf{u} =_! p.\langle \delta_{p,x} \rangle^\mathsf{u} \qquad \langle \delta_{p,x} \rangle^\mathsf{u} = \bigcirc_{t \in \delta_{p,x}}^\mathsf{u} \langle t \rangle_x^\mathsf{u}$$

$$\langle (p, x, w, q) \rangle_x^\mathsf{u} = s_\mathsf{u} \coloneqq q.w_0^\mathsf{u} \ldots w_n^\mathsf{u}$$

$$\langle (p, \mathrm{nx}, q) \rangle_x^\mathsf{h} = (gn =_! -.gn \coloneqq +.\$^\mathsf{h}.@.x^\mathsf{h}) \vee^\mathsf{h} (gn =_! +.s_\mathsf{h} \coloneqq q.c_\mathsf{h} \coloneqq \mathrm{nx}.x^\mathsf{h})$$

$$\langle (p, \mathrm{nx}, q) \rangle_x^\mathsf{v} = s_\mathsf{v} \coloneqq q.c_\mathsf{v} \coloneqq \mathrm{nx}.x^\mathsf{v}$$

$$@ = c_1 \coloneqq \mathrm{st} \ldots c_\mathsf{h} \coloneqq \mathrm{st}$$

$$\mathrm{H}^\mathsf{u} = \mathrm{H}^{\mathsf{u}-1}.\bigvee_{p \in Q}^\mathsf{u} f_\mathsf{u} \coloneqq p.s_\mathsf{u} \coloneqq p$$

The initial term for the game is simply $\mathrm{H}^{\mathsf{h}-1}.\underline{\$^\mathsf{h}}.@$. The terminals $\langle \to \mathrm{st} \rangle^\mathsf{u}$ and $\langle \to \mathrm{nx} \rangle^\mathsf{u}$ used in the main paper refer to the assertions $c_\mathsf{u} =_! \mathrm{st}$ and $c_\mathsf{u} =_! \mathrm{nx}$. At context switches in urgency h, Eve also needs to "guess" whether the second stack has been generated. In the case where it has not yet been generated, the correct choice generates it. In the case where it has already been generated, the correct choice triggers a context switch in the usual way.

M. Denotational Semantics

We show how to define a denotational semantics based on our axiomatization. What we find interesting is that, with the axiomatization at hand, the denotational semantics is a derived construct: the semantic domain and the interpretation of function symbols are induced by the axiomatization, yet the semantics is guaranteed to be fully abstract wrt. contextual equivalence resp. its specialized variant. The creativity that is saved in the definition of the semantics of course had to be invested up front when coming up with the axiomatization. We found it easier to study an axiomatization than a denotational semantics, because the problem is narrowed down to understanding the interplay between operators as opposed to coming

up with a freely chosen semantic domain. We recall the basics of denotational semantics before turning to the details.

A *denotational semantics* for our programming language is a pair $((\mathcal{D}, \subseteq), \mathcal{I})$ consisting of a complete partial order $(\mathcal{D}, \subseteq)$ of semantic elements and an interpretation $\mathcal{I} : F \to \mathcal{D}^\omega \to \mathcal{D}$ that assigns to each function symbol $f \in F$ in our language a monotonic function $f^{\mathcal{I}} : \mathcal{D}^{ar(f)} \to \mathcal{D}$ of the expected arity. The function symbols F are Σ, $\{\mathsf{skip}, \mathsf{err}, .\}$, and choices of arbitrary arity with urgency 1 to h. We lift the interpretation to all terms p and assign them an element $\mathcal{D}(p) \in \mathcal{D}$, called the denotational semantics of the term. For recursion-free terms, the lifting is purely compositional:

$$\mathcal{D}(a) = a^{\mathcal{I}} \qquad \mathcal{D}(p.q) = \mathcal{D}(p).^{\mathcal{I}}\mathcal{D}(q),$$

and similar for the other function symbols. For the non-terminals (N, E), this allows us to understand the defining equations as a monotonic function

$$E^{\mathcal{I}} : (N \to \mathcal{D}) \to N \to \mathcal{D}.$$

The least solution of this function is the denotational semantics of the non-terminals: $\mathcal{D}(A) = [\mathit{lfp}.E^{\mathcal{I}}](A)$ for every $A \in N$. This is the missing case to define the semantics of arbitrary program terms again in a compositional way.

We focus on the denotational semantics induced by the axiomatic congruence. The development for the O-specialized axiomatic congruence with O right-separating is the same. If O is not right-separating, we cannot give a guarantee that the resulting semantics will be fully abstract. The *denotational semantics induced by* $\equiv$ is $((\mathcal{D}_\equiv, \sqsubseteq), \mathcal{I}_\equiv)$. The set of semantic elements is $\mathcal{D}_\equiv = \mathbf{T}/_\equiv$, we factorize the set of terms along the axiomatic congruence. The complete partial order on these congruence classes is the one given by the axiomatic precongruence. It is guaranteed to be well-defined due to the precongruence. It is guaranteed to stabilize in an ordinal by the fact that chains are well-ordered sets. The interpretation of the function symbols is as expected:

$$a^{\mathcal{I}_\equiv} = [a]_\equiv \qquad [p]_\equiv .^{\mathcal{I}_\equiv} [q]_\equiv = [p.q]_\equiv.$$

Well-definedness holds because $\equiv$ is a congruence, monotonicity holds because $\sqsubseteq$ is a precongruence. The semantics is fully abstract wrt. contextual equivalence, $\mathcal{D}(p) = \mathcal{D}(q)$ iff $p \preceq q$, which is merely a reformulation of Theorem 1. We can define other fully abstract semantics by introducing representative systems on the congruence classes, for example based on normal forms.