Progress, Justness and Fairness in Modal μ-Calculus Formulae

Myrthe S. C. Spronck ✉ (ID)
Eindhoven University of Technology, The Netherlands

Bas Luttik ✉ (ID)
Eindhoven University of Technology, The Netherlands

Tim A. C. Willemse ✉ (ID)
Eindhoven University of Technology, The Netherlands

──── **Abstract** ────────────────────────────

When verifying liveness properties on a transition system, it is often necessary to discard spurious violating paths by making assumptions on which paths represent realistic executions. Capturing that some property holds under such an assumption in a logical formula is challenging and error-prone, particularly in the modal μ-calculus. In this paper, we present template formulae in the modal μ-calculus that can be instantiated to a broad range of liveness properties. We consider the following assumptions: progress, justness, weak fairness, strong fairness, and hyperfairness, each with respect to actions. The correctness of these formulae has been proven.

1 Introduction

Formal verification through model checking requires a formalisation of the properties of the modelled system as formulae in some logic, such as LTL [33], CTL [18] or the modal μ-calculus [30]. In this paper, we focus on the modal μ-calculus, a highly expressive logic used in established model checkers such as mCLR2 [11] and CADP [20].

A frequently encountered problem when checking liveness properties is that spurious violations are found, such as paths on which some components never make progress. Often, such paths do not represent realistic executions of the system. It is then a challenge to restrict verification to those paths that do represent realistic system executions. For this, we use completeness criteria [22, 23]: predicates on paths that say which paths are to be regarded as realistic runs of the system. These runs are called complete runs. Examples of completeness criteria are progress, justness and fairness.

It turns out that writing a modal μ-calculus formula for a property being satisfied under a completeness criterion is non-trivial. Since the μ-calculus is a branching-time logic, we cannot separately formalise when a path is complete and when it satisfies the property, and then combine the two formalisations with an implication. Instead, a more intricate integration of both aspects of a path is needed. Our aim is to achieve such an integration for a broad spectrum of liveness properties and establish the correctness of the resulting formulae. To this end, we shall consider a template property that can be instantiated to a plethora of liveness properties and, in particular, covers all liveness property patterns of [17]. Then, we present modal μ-calculus formulae integrating the completeness criteria of progress, justness, weak fairness, strong fairness, and hyperfairness with this template property.

As discussed in [24], for the formulation of realistic completeness criteria it is sometimes necessary to give special treatment to a set of blocking actions, i.e., actions that require cooperation of the environment in which the modelled system operates. Our template formulae are therefore parameterised with a set of blocking actions. We shall see that, given a set of blocking actions, there are two different interpretations of hyperfairness; we call these weak and strong hyperfairness.

Regarding our presented formulae, the progress formula is similar to those commonly used for liveness properties even when completeness is not explicitly considered. Our formulae for justness, weak fairness and weak hyperfairness only subtly differ from each other. We characterise the similarities these three share and give a generic formula that can be adapted to represent all completeness criteria that meet these conditions. Lastly, we observe that strong fairness and strong hyperfairness do not meet these conditions. We give alternative formulae that are significantly more complex. Whether more efficient formulae for these completeness criteria exist remains an open problem.

Modal μ-calculus formulae are often hard to interpret. Accordingly, it is not trivial to see that our formulae indeed express the integration of liveness properties with completeness criteria. We have therefore included elaborate correctness proofs in the appendices.

Our work is essentially a generalisation along two dimensions (viz., the completeness criterion and the liveness property) of the works of [35] and [7, 37]. In [35], the tool PASS is presented for automatically translating common property patterns into modal μ-calculus formulae. Some of those patterns integrate an assumption that excludes paths deemed unrealistic, but since the exact assumption is not stated separately, we cannot make a formal comparison with our approach. In [7], a formula for justness is presented, covering one of the properties we cover. This formula forms the basis for our justness, weak fairness and weak hyperfairness formulae. Our formulae for strong fairness and strong hyperfairness are in part inspired by the formula for termination under strong fairness presented in [37].

The organisation of this paper is as follows. In Section 2 we recap the relevant definitions on labelled transition systems, as well as the syntax and semantics of the modal μ-calculus. In Section 3, we motive our work with an example, and in Section 4 we give the completeness criteria we cover in this paper. In Section 5, we formally identify the class of liveness properties we study and relate it to a popular class of properties. Our template formulae are presented in Section 6, combining the completeness criteria from Section 4 with the property template from Section 5. We give a small application example in Section 7 and discuss the scope of our work in Section 8. Finally, we give our conclusions in Section 9.

2   Preliminaries

We represent models as labelled transition systems (LTSs). In this section, we briefly introduce the relevant definitions on LTSs, as well as the modal μ-calculus.

2.1   Labelled Transition Systems

▶ **Definition 1.** *An LTS is a tuple $M = (\mathcal{S}, s_{init}, Act, Trans)$ where*
- *$\mathcal{S}$ is a set of states,*
- *$s_{init} \in \mathcal{S}$ is the initial state,*
- *Act is a set of action labels, also referred to as the alphabet of the LTS, and*
- *$Trans \subseteq \mathcal{S} \times Act \times \mathcal{S}$ is a transition relation.*

In this paper, we only consider finite LTSs, such as the kind used in finite-state model checking. In particular, our formulae are proven correct under the assumption that *Act*

is finite. We write $s \xrightarrow{a} s'$ as shorthand for $(s, a, s') \in \textit{Trans}$, and for a given transition $t = (s, a, s')$ we write $src(t) = s$, $act(t) = a$ and $trgt(t) = s'$.

For the definitions below, we fix an LTS $M = (\mathcal{S}, s_{init}, \textit{Act}, \textit{Trans})$.

▶ **Definition 2.** *A path is an (alternating) sequence $\pi = s_0 t_1 s_1 t_2 \ldots$ of states $s_0, s_1, \ldots \in \mathcal{S}$ and transitions $t_1, t_2, \ldots \in \textit{Trans}$. A path must start with a state, and must be either infinite, or end in a state. In the latter case, the end of the path is referred to as the final state. For all $i \geq 0$, t_{i+1} must satisfy $src(t_{i+1}) = s_i$ and $trgt(t_{i+1}) = s_{i+1}$.*

We sometimes refer to transitions on a path as steps. We say an action occurs on a path if a transition labelled with that action is on the path. We call a path on which no action in some set α occurs an α-*free* path. One path can be appended to another: let $\pi' = s'_0 t'_1 s'_1 \ldots t'_n s'_n$ and $\pi'' = s''_0 t''_1 s''_1 \ldots$, where π' must be finite and π'' may be finite or infinite. Then the path π defined as π'' appended to π' is written as $\pi = \pi' \cdot \pi'' = s'_0 t'_1 s'_1 \ldots t'_n s'_n t''_1 s''_1 \ldots$. This is only allowed when $s'_n = s''_0$.

▶ **Definition 3.** *We say that:*
- *A transition $t \in \textit{Trans}$ is* enabled *in a state $s \in \mathcal{S}$ if, and only if, $src(t) = s$.*
- *An action $a \in \textit{Act}$ is* enabled *in a state $s \in \mathcal{S}$ if, and only if, there exists a transition $t \in \textit{Trans}$ with $act(t) = a$ that is enabled in s.*
- *An action $a \in \textit{Act}$ is* perpetually enabled *on a path π if a is enabled in every state of π.*
- *An action $a \in \textit{Act}$ is* relentlessly enabled *on a path π if every suffix of π contains a state in which a is enabled.*
- *A state without enabled actions is called a* deadlock *state.*

Every action that is perpetually enabled on a path is also relentlessly enabled on that path.

2.2 Modal μ-Calculus

The modal μ-calculus is given in [30]. Our presentation of the logic is based on [8, 9, 10, 27].

The syntax of the modal μ-calculus is described by the following grammar, in which a ranges over the set of actions *Act*, and X ranges over a set of formal variables *Var*.

$$\phi, \psi ::= \textit{ff} \mid X \mid \neg\phi \mid \phi \vee \psi \mid \langle a \rangle \phi \mid \mu X.\phi$$

Here *ff* is false; $\neg$ represents negation; $\vee$ is disjunction; $\langle\ \rangle$ is the diamond operator; and μ is the least fixpoint operator. We say that $\mu X.\phi$ binds X in ϕ. Variables that are unbound in a formula are free, and a formula without free variables is closed.

A modal μ-calculus formula ϕ must both adhere to this grammar and be *syntactically monotonic*, meaning that for every occurrence of $\mu X.\psi$ in ϕ, every free occurrence of X in ψ must always be preceded by an even number of negations.

We give the semantics of a modal μ-calculus formula ϕ with respect to an arbitrary LTS $M = (\mathcal{S}, s_{init}, \textit{Act}, \textit{Trans})$ and environment $e : \textit{Var} \to 2^{\mathcal{S}}$.

$$\llbracket \textit{ff} \rrbracket_e^M = \emptyset \qquad\qquad \llbracket \phi \vee \psi \rrbracket_e^M = \llbracket \phi \rrbracket_e^M \cup \llbracket \psi \rrbracket_e^M$$

$$\llbracket X \rrbracket_e^M = e(X) \qquad\qquad \llbracket \langle a \rangle \phi \rrbracket_e^M = \left\{ s \in \mathcal{S} \mid \exists_{s' \in \mathcal{S}}.s \xrightarrow{a} s' \wedge s' \in \llbracket \phi \rrbracket_e^M \right\}$$

$$\llbracket \neg\phi \rrbracket_e^M = \mathcal{S} \setminus \llbracket \phi \rrbracket_e^M \qquad\qquad \llbracket \mu X.\phi \rrbracket_e^M = \bigcap \left\{ \mathcal{S}' \subseteq \mathcal{S} \mid \mathcal{S}' \supseteq \llbracket \phi \rrbracket_{e[X := \mathcal{S}']}^M \right\}$$

In contexts where the model is fixed, we drop the M from $\llbracket \phi \rrbracket_e^M$. Additionally, we drop e when the environment does not affect the semantics of the formula, e.g. with closed formulae.

We use conjunction, $\wedge$, and implication, $\Rightarrow$, as the usual abbreviations. We also add several abbreviations: $tt = \neg f\!f$ for true; $[a]\phi = \neg\langle a\rangle\neg\phi$ for the box operator; and $\nu X.\phi = \neg\mu X.(\neg\phi[X := \neg X])$ for the greatest fixpoint.

To express formulae more compactly, we extend our syntax to allow regular expressions over finite sets of actions to be used in the box and diamond operators. Since we limit this to finite sets of actions, the syntactical extension does not increase the expressivity of the logic, it merely simplifies the presentation. This is a common extension of the μ-calculus syntax, for instance shown in [27], based on the operators defined for PDL [19]. We overload the symbol for a single action to also represent the singleton set containing that action. We use union, intersection, set difference, and set complement to describe sets of actions as usual. Regular expressions over sets of actions, henceforth referred to as *regular formulae*, are defined by the following grammar:

$$R, Q ::= \varepsilon \mid \alpha \mid R \cdot Q \mid R + Q \mid R^{\star}$$

The empty sequence is represented by ε, and α ranges over sets of actions. The symbol $\cdot$ represents concatenation, $+$ the union of formulae, and * is closure under repetition.

We define the meaning of the diamond operator over the new regular formulae as abbreviations of standard modal μ-calculus formulae:

$$\langle\varepsilon\rangle\phi = \phi \qquad\qquad \langle\alpha\rangle\phi = \bigvee_{a\in\alpha}\langle a\rangle\phi \qquad\qquad \langle R\cdot Q\rangle\phi = \langle R\rangle\langle Q\rangle\phi$$

$$\langle R + Q\rangle\phi = \langle R\rangle\phi \vee \langle Q\rangle\phi \qquad \langle R^{\star}\rangle\phi = \mu X.(\langle R\rangle X \vee \phi)$$

The box operator is defined dually. We say a path π *matches* a regular formula R if the sequence of actions on π is in the language of R.

3 Motivation

When analysing algorithms and systems, there are many different properties which may need to be checked. For instance, when model checking mutual exclusion algorithms we want to check linear properties such as mutual exclusion and starvation freedom, but also branching properties such as invariant reachability of the critical section. The modal μ-calculus, which subsumes even CTL*, is able to express all these properties and more, and is therefore used in toolsets such as mCLR2 [11] and CADP [20].

An issue that is frequently encountered when checking liveness properties in particular, is that the model admits executions that violate the property but do not represent realistic executions of the real system. For example, models of algorithms that contain a busy waiting loop usually admit executions where processes do nothing except wait. Infinite loops can also be introduced by abstractions of reality, such as modelling a loop to represent an event that occurs an arbitrary, but finite, number of times. Counterexamples that are due to such modelling artefacts obscure whether the property is satisfied on all realistic executions. The problem we address in this paper is how to avoid such counterexamples and check properties only on realistic executions. We illustrate the problem with an example, which we also employ as a running example throughout this paper.

▶ **Example 4.** Consider the coffee machine modelled in Figure 1. When a user places an *order* for one or more cups of coffee, they are required to scan their payment *card*. If the user prefers using coinage, they switch the machine to its alternate mode (*to_cash*), and then pay in *cash*. In the alternate mode, the machine can be switched back using *to_card*. After
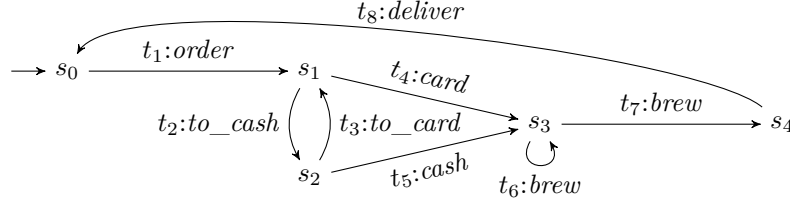
■ **Figure 1** The LTS for the running example.

payment, the machine will *brew* the cup(s) of coffee. This is modelled as a non-deterministic choice between a looping and a final *brew* action, since at least one cup was ordered. Finally, the coffee is *deliver*ed and the machine awaits the next order.

We consider three example properties.

1. *Single order*: whenever an *order* is made, there may not be a second *order* until a *deliver* has taken place, $[Act^\star \cdot order \cdot \overline{deliver}^\star \cdot order]ff$.
2. *Inevitable delivery*: whenever an *order* is made, there will inevitably be an occurrence of *deliver*, $[Act^\star \cdot order]\mu X.(\langle Act\rangle tt \wedge [\overline{deliver}]X)$.
3. *Possible delivery*: it is invariantly possible to eventually execute the *deliver* action, $[Act^\star]\langle Act^\star \cdot deliver\rangle tt$.

The described problem occurs with *inevitable delivery*: $s_0 t_1 s_1 t_4 (s_3 t_6)^\omega$ is a violating path, on which infinitely many cups are part of the same order. Similarly, $s_0 t_1 (s_1 t_2 s_2 t_3)^\omega$ violates the property because the user never decides on a payment method. The first counterexample represents an impossible scenario, and the second gives information on problematic user behaviour but tells us little about the machine itself.

The kind of spurious counterexamples discussed in the example above primarily occur when checking liveness properties. We therefore focus on liveness properties, such as *inevitable delivery*, in this paper. We will briefly discuss safety properties in Section 8.

There are ad-hoc solutions to exclude unrealistic counterexamples, e.g. altering the model to remove the unrealistic executions, or tailoring the formula to exclude specific problematic counterexamples [26]. Such ad-hoc solutions are undesirable because they clutter the model or the formula, and are therefore error-prone. We aim for a more generic solution, of which the correctness can be established once and for all. Such a generic solution requires, on the one hand, a general method to distinguish between realistic and unrealistic executions, and, on the other hand, a general class of liveness properties.

A general method to distinguish between realistic and unrealistic executions is provided by *completeness criteria* [22, 23], i.e., predicates on paths that label some as complete and all others as incomplete. If a property is satisfied on all complete paths, it is satisfied under the given completeness criterion. Completeness criteria give us a model-independent way to determine which paths are unrealistic, and therefore a generic solution to the stated problem. Depending on the property and the model, we may prefer a different completeness criterion. We therefore consider several criteria instead of fixing one specific criterion. These completeness criteria are discussed in Section 4.

To find a general class of liveness properties, we take the *property specification patterns* (PSP) of [17] as a starting point. Since the modal μ-calculus as presented in Section 2.2 supports references to action occurrences but not state information, we specifically interpret these patterns on action occurrences. Our first contribution, in Section 5, will be to characterise a class of liveness properties that subsumes all liveness properties expressible in

PSP. Our second and main contribution is then presented in Section 6, where we combine the identified completeness criteria with our class of liveness properties, yielding template formulae for each combination.

4 Completeness Criteria

It is often assumed, sometimes implicitly, that as long as a system is capable of executing actions, it will continue to do so [25]. One could consider this the "default" completeness criterion, also known as *progress* [22]; it says that only paths that are infinite or end in a deadlock state model complete runs and are hence complete paths. We first present a modified version of the progress assumption that allows some actions to be blocked by the environment. We then define the other completeness criteria considered in this paper. As already remarked in the previous section, the modal μ-calculus is most suited to reasoning about action occurrences. Hence, we focus on completeness criteria defined on action labels. For more general definitions on sets of transitions, see [25].

4.1 Progress with Blocking Actions

In [24], it is argued that it is useful to consider some actions of an LTS as blocking. A blocking action is an action that depends on participation by the environment of the modelled system. Consequently, even when such an action is enabled in a state because the system is willing to perform it, it may not be possible for the action to occur because the environment is uncooperative. In this paper, we refer to the set of blocking actions as $\mathcal{B} \subseteq Act$, and the set of non-blocking actions as $\overline{\mathcal{B}} = Act \setminus \mathcal{B}$. Which actions are in $\mathcal{B}$ is a modelling choice.

The default progress assumption can be adapted to account for blocking actions [21, 25].

▶ **Definition 5.** *A state $s \in \mathcal{S}$ is a $\mathcal{B}$-locked state if, and only if, all actions enabled in s are in $\mathcal{B}$. A path π is $\mathcal{B}$-progressing if, and only if, it is infinite or ends in a $\mathcal{B}$-locked state.*

We refer to the assumption that only $\mathcal{B}$-progressing paths represent complete executions as $\mathcal{B}$-progress. The "default" completeness criterion is equivalent to $\emptyset$-progress.

▶ **Example 6.** Consider Figure 1. Here, *order* is an environment action, since it involves the user. If we do not assume that there will always be a next user, we should add *order* to $\mathcal{B}$. In some cases, we may want to consider the possibility that the machine is broken and not capable of producing coffee. In those cases, we should add *brew* to $\mathcal{B}$. Our choice of $\mathcal{B}$ affects which paths are progressing: $s_0 t_1 s_1 t_4 s_3$ is not $\emptyset$-progressing, but it is $\{brew\}$-progressing.

All completeness criteria we discuss in this paper are parameterised with a set of blocking actions. The justness and fairness assumptions discussed in the remainder of this section label paths as incomplete if certain actions do not occur. Since it can never be assumed that the environment supports the occurrence of blocking actions, we do not want justness and fairness to label paths as incomplete due to the non-occurrence of blocking actions.

For readability the prefix $\mathcal{B}$- will sometimes be dropped from the names of the completeness criteria and their acronyms. From this point, we will always discuss completeness criteria with respect to a set of blocking actions.

4.2 Justness

Justness [21, 25] is a natural extension of progress to exclude infinite paths instead of finite paths. The idea is that in addition to the system as a whole progressing, individual

components in that system should also be able to make progress unless they are prevented from doing so by other components. It is a weaker, and hence frequently more justifiable, assumption than the fairness assumptions we cover in the next section. In its original presentation, justness is defined with respect to sets of transitions. Which components contribute to a transition and how they contribute to them determines which transitions interfere with each other. We here consider justness defined with respect to actions instead, based on [7]. We do not go into how it is determined which actions interfere with each other here. For discussions on this topic and when the two definitions coincide, see [6, 7, 21].

Intuitively, justness of actions says that if an action a is enabled at some point of a path, then eventually some action that can interfere with the occurrence of a must occur in that path. That action may be a itself. In order to formalise the concept of interference, we require the concept of a *concurrency relation on actions*, $\smile\!\bullet$.

▶ **Definition 7.** *Relation* $\smile\!\bullet\, \subseteq Act \times Act$ *is a* concurrency relation on actions *if, and only if:*
1. $\smile\!\bullet$ *is irreflexive, and*
2. *for all* $a \in Act$, *if* π *is a path from a state* s *in which* a *is enabled to a state* $s' \in \mathcal{S}$ *such that* $a \smile\!\bullet\, b$ *for all actions* b *occurring in* π, *then* a *is enabled in* s'.

We write $\not\smile\!\bullet$ for the complement of $\smile\!\bullet$. Note that $\smile\!\bullet$ may be asymmetric.

Read $a \smile\!\bullet\, b$ as "a is concurrent with b", and $a \not\smile\!\bullet\, b$ as "b interferes with a" or "b eliminates a". A labelled transition system can be extended with a concurrency relation on actions, which produces a *labelled transition system with concurrency* (LTSC).

We here present the definition for justness of actions with blocking actions.

▶ **Definition 8.** *A path* π *satisfies* $\mathcal{B}$-*justness of actions* ($\mathcal{B}$-JA) *if, and only if, for each action* $a \in \overline{\mathcal{B}}$ *that is enabled in some state* s *in* π, *an action* $a' \in Act$ *occurs in the suffix* π' *of* π *starting in* s *such that* $a \not\smile\!\bullet\, a'$.

▶ **Example 9.** Consider Figure 1, specifically the path $s_0 t_1 (s_1 t_2 s_2 t_3)^\omega$. On this path the user keeps switching the mode of the machine, without paying. To see if this path satisfies $\emptyset$-JA, we need a concrete $\smile\!\bullet$. Consider a $\smile\!\bullet$ such that *card* $\not\smile\!\bullet$ *to_cash*, *cash* $\not\smile\!\bullet$ *to_card*, and $a \not\smile\!\bullet\, a$ for all action labels a. These are all required for $\smile\!\bullet$ to be a valid concurrency relation. This is because by Definition 7, $\smile\!\bullet$ must be irreflexive, and when an action is enabled it must remain enabled on any path on which no interfering action occurs. Since *card* is enabled in s_1 but not s_2, it must be the case that *card* $\not\smile\!\bullet$ *to_cash*. Similarly, we must have *cash* $\not\smile\!\bullet$ *to_card*. With such a concurrency relation, the path satisfies $\emptyset$-JA since every action that is enabled is subsequently eliminated. In this LTS, there is no valid choice of $\smile\!\bullet$ that makes this path violate $\emptyset$-JA. However, if we modify Figure 1 by replacing both *card* and *cash* with the action *pay*, then Definition 7 does not enforce that *to_cash* and *to_card* interfere with the actions on t_4 and t_5, since *pay* is enabled in both s_1 and s_2. We can choose whether *pay* $\smile\!\bullet$ *to_cash* and *pay* $\smile\!\bullet$ *to_card*. If *pay* is concurrent with both, then the path $s_0 t_1 (s_1 t_2 s_2 t_3)^\omega$ violates $\emptyset$-JA. If either interferes with *pay*, then the path satisfies $\emptyset$-JA.

4.3 Fairness

There are situations where we want to exclude a larger set of infinite paths than those excluded by justness, or where we do not have a concurrency relation. For this, we can use what are called *fairness assumptions* in the literature. These are a class of predicates on paths that distinguish between *fair* and *unfair* infinite paths. It is assumed that only the fair paths are complete. For an overview of many common fairness assumptions, see [25]. In this paper, we consider weak fairness of actions, strong fairness of actions, and (weak and strong)

hyperfairness of actions. Each of the assumptions we discuss has the general shape, adapted from [3], "if it is sufficiently often possible for an action to occur, it will occur sufficiently often". What it means for an action to be "sufficiently often possible" and "occur sufficiently often" depends on the exact assumption.

We first discuss *weak fairness of actions*, which says that actions that are always enabled must eventually occur. It is one of the most commonly discussed fairness assumptions. We define weak fairness of actions formally, with respect to a set of blocking actions $\mathcal{B}$.

▶ **Definition 10.** *A path π satisfies $\mathcal{B}$-weak fairness of actions ($\mathcal{B}$-WFA) if, and only if, for every suffix π' of π, every action $a \in \overline{\mathcal{B}}$ that is perpetually enabled in π' occurs in π'.*

▶ **Example 11.** Consider again Figure 1, with *card* and *cash* both replaced by *pay*. Then the path $s_0 t_1 (s_1 t_2 s_2 t_3)^\omega$ violates $\emptyset$-WFA, since *pay* is perpetually enabled in a suffix of this path without occurring. If there are two separate actions for paying with cash or card, the path satisfies $\emptyset$-WFA because no actions are perpetually enabled in any suffix.

Next, *strong fairness of actions* says that on a path, all actions that are enabled infinitely often, must occur infinitely often. Formally, we define strong fairness of actions as:

▶ **Definition 12.** *A path π satisfies $\mathcal{B}$-strong fairness of actions ($\mathcal{B}$-SFA) if, and only if, for every suffix π' of π, every action $a \in \overline{\mathcal{B}}$ that is relentlessly enabled in π' occurs in π'.*

Strong fairness is a stronger assumption than weak fairness, since it classifies more paths as incomplete. This follows from perpetual enabledness implying relentless enabledness.

▶ **Example 13.** The path $s_0 t_1 (s_1 t_2 s_2 t_3)^\omega$ in Figure 1 satisfies $\emptyset$-WFA since there are no perpetually enabled actions in any suffix of the path. However, *cash* is relentlessly enabled in suffixes of this path, and yet does not occur. Hence, this path violates $\emptyset$-SFA.

Finally, we discuss *hyperfairness of actions*. Informally, it says that on all fair paths, every action that can always become enabled must occur infinitely often. The idea is that if there is always a reachable future where the action occurs, then it is merely unlucky if the action does not occur infinitely often. The concept of hyperfairness is introduced and named in [4]. For our presentation of hyperfairness, we use the generalisation from [31]. We first formalise what it means that an action "can become" enabled, by defining *reachability*.

▶ **Definition 14.** *We say that:*
- *A state $s \in \mathcal{S}$ is $\mathcal{B}$-reachable from some state $s' \in \mathcal{S}$ if, and only if, there exists a $\mathcal{B}$-free path starting in s' that ends in s.*
- *An action $a \in Act$ is $\mathcal{B}$-reachable from some state $s \in \mathcal{S}$ if, and only if, there exists a state $s' \in \mathcal{S}$ that is $\mathcal{B}$-reachable from s and in which a is enabled.*
- *A state $s \in \mathcal{S}$ or action $a \in Act$ is perpetually $\mathcal{B}$-reachable on a path π if, and only if, it is $\mathcal{B}$-reachable from every state of π.*
- *A state $s \in \mathcal{S}$ or action $a \in Act$ is relentlessly $\mathcal{B}$-reachable on a path π if, and only if, every suffix of π contains a state from which it is $\mathcal{B}$-reachable.*

From the intuitive description of hyperfairness, it is clear it is a variant of weak or strong fairness with reachability instead of enabledness, giving us two possible definitions of hyperfairness. We name the two interpretations weak hyperfairness and strong hyperfairness respectively. Both interpretations of hyperfairness are reasonable, and in fact when not considering blocking actions, they coincide [31]. However, this is not the case when blocking actions are included in the definitions. We therefore consider both variants.

▶ **Definition 15.** *A path π satisfies* weak $\mathcal{B}$-hyperfairness of actions ($\mathcal{B}$-WHFA) *if, and only if, for every suffix π' of π, every action $a \in \overline{\mathcal{B}}$ that is perpetually $\mathcal{B}$-reachable in π' occurs in π'.*

▶ **Definition 16.** *A path π satisfies* strong $\mathcal{B}$-hyperfairness of actions ($\mathcal{B}$-SHFA) *if, and only if, for every suffix π' of π, every action $a \in \overline{\mathcal{B}}$ that is relentlessly $\mathcal{B}$-reachable in π' occurs in π'.*

Since enabledness implies reachability, WHFA is stronger than WFA, and SHFA is stronger than SFA. Perpetually reachability implies relentless reachability, so SHFA is also stronger than WHFA. However, as the next examples will show, SFA and WHFA are incomparable.

▶ **Example 17.** The impact of hyperfairness can clearly be seen when non-determinism is used. Consider the path $s_0 t_1 s_1 t_4 (s_3 t_6)^\omega$ in Figure 1. This path satisfies $\emptyset$-SFA, since the only action that is relentlessly enabled on this path, *brew*, also occurs infinitely often. However, as long as *deliver* $\notin \mathcal{B}$ and *brew* $\notin \mathcal{B}$, this path does not satisfy $\mathcal{B}$-WHFA or $\mathcal{B}$-SHFA: *deliver* is $\mathcal{B}$-reachable from s_3, and therefore is perpetually and relentlessly $\mathcal{B}$-reachable in a suffix of this path, but does not occur. We here see $\mathcal{B}$-SFA does not imply $\mathcal{B}$-WHFA.

▶ **Example 18.** In Figure 1, consider $s_0 t_1 (s_1 t_2 s_2 t_3)^\omega$ with $\mathcal{B} = \{order, to\_cash, to\_card\}$. This path satisfies $\mathcal{B}$-WHFA because *card* and *cash* are only $\mathcal{B}$-reachable from s_1 and s_2 respectively. They are not perpetually $\mathcal{B}$-reachable in any suffix of this path, therefore $\mathcal{B}$-WHFA is satisfied. However, they are relentlessly $\mathcal{B}$-reachable, so $\mathcal{B}$-SHFA is violated. This demonstrates that $\mathcal{B}$-WHFA and $\mathcal{B}$-SFHA do not coincide when blocking actions are considered. The actions *card* and *cash* are also relentlessly $\mathcal{B}$-enabled, so $\mathcal{B}$-SFA is also violated. Hence, $\mathcal{B}$-WHFA does not imply $\mathcal{B}$-SFA.

5   A Generalisation of the Property Specification Liveness Patterns

Dwyer, Avrunin and Corbett observed that a significant majority of properties that are used in practice can be fit into a set of property specification patterns [17]. These patterns consist of a *behaviour* that must be satisfied and a *scope* within a path that delimits where the behaviour must be satisfied. We recall the behaviours and scopes presented in [17] in Appendix A. We focus on expressing properties that are captured by PSP.

Of all behaviours considered in [17], only *existence*, *existence at least*, *response* and *chain response* represent pure liveness properties. The *global* and *after* scopes, when combined with any of these four behaviours, give liveness properties. We argue why only these patters of PSP represent pure liveness properties in Appendix A. All other scopes result in safety properties or properties that combine safety and liveness. Of those, we cover the *until* and *after-until* scopes, since we can incorporate those into our formulae with little difficulty.

For behaviours, *existence at least* says some action in a set S_r must occur at least k times in the scope; when $k = 1$ we call this *existence*. The *response* behaviour requires that whenever an action in a set S_q occurs, it must be followed by the occurrence of an action in S_r. When chains of action occurrences are used instead of individual action occurrences, this is called *chain response*. For the scopes, *global* refers to the full path and *after* to the path after the first occurrence of an action in a set S_a. The *until* scope refers to the path before the first occurrence of an action in a set S_b, or the full path if no such action occurs. Finally, *after-until* combines *after* and *until*, referring to every subpath of the path that starts after any occurrence of an action in S_a and ends before the following occurrence of an action in S_b. If no action in S_b occurs, the behaviour must still be satisfied after S_a.

▶ **Example 19.** Consider again the properties we presented in Example 4. *Single order* is *absence after-until*, with $S_a = \{order\}$, $S_b = \{deliver\}$ and $S_r = \{order\}$. *Inevitable delivery* is *global response* with $S_q = \{order\}$ and $S_r = \{deliver\}$. *Possible delivery* does not fit into the patterns on occurrences of actions, since it contains a requirement on states, specifically that the state admits a path on which *delivery* occurs.

We want to create formulae for all 16 combinations of the selected behaviours and scopes. To make our results more compact and generic, we first generalise these 16 patterns into a single template property. This template works by describing the shape of a violating path for a property that fits one of these patterns. Intuitively, this shape is: "after the occurrence of ρ, there are no occurrences of α_f up until the (optional) occurrence of α_e". For our template formulae to be syntactically correct, it is important that ρ is a regular formula, describing the prefix that a violating path must have, whereas α_f and α_e are sets of actions. The actions in α_f are those that are forbidden from occurring after ρ on a violating path, whereas the actions in α_e indicate the end of the scope in which α_f may not occur.

We formalise this template as follows:

▶ **Definition 20.** *A path π is $(\rho, \alpha_f, \alpha_e)$-violating if, and only if, there exist π_{pre} and π_{suf} such that:*

1. *$\pi = \pi_{pre} \cdot \pi_{suf}$, and*
2. *π_{pre} matches ρ, and*
3. *π_{suf} satisfies at least one of the following conditions:*
   a. *π_{suf} is α_f-free, or*
   b. *π_{suf} contains an occurrence of an action in α_e, and the prefix of π_{suf} before the first occurrence of an action in α_e is α_f-free.*

For readability, we frequently refer to $(\rho, \alpha_f, \alpha_e)$-violating paths as violating paths. We sometimes summarise condition 3 as "π_{suf} is α_f-free up until the first occurrence of α_e". See Figure 2 for an illustration of what types of paths are considered violating.

Figure 2 The four types of $(\rho, \alpha_f, \alpha_e)$-violating paths: finite or infinite, and without or with α_e. Always, it has a prefix matching ρ and is α_f-free up until the first occurrence of an action in α_e.

All 16 patterns can indeed be represented by the non-existence of $(\rho, \alpha_f, \alpha_e)$-violating paths, albeit some more directly than others. It turns out that ρ, α_f and α_e can mostly be determined separately for behaviour and scope. For these patterns, α_f is only affected by behaviour and α_e only by scope. However, we must split up the regular formula ρ into a behaviour component, ρ_b, and scope component, ρ_s, such that $\rho = \rho_s \cdot \rho_b$. See Table 1a and Table 1b for how the variables should be instantiated for the four scopes and three of the four behaviours. For a compact representation, we use $\sum$ to generalise the union operator on regular formulae $(+)$. We also use x^i to represent i concatenations of x, where $x^0 = \varepsilon$.

We do not include *chain response* in Table 1b, since it does not fit into a single formula. However, it is possible to represent *chain response* as several *response* formulae placed in conjunction with each other. We include an example of this in Appendix B.

6   Template Formulae

In this section, we present the modal μ-calculus formulae representing the non-existence of a violating path, as defined in Section 5, that satisfies one of the completeness criteria from

Table 1 Variable instantiation for templates.

(a) For scopes.

Scope	ρ_s	α_e
Global	ε	$\emptyset$
Until	ε	S_b
After	$\overline{S_a}^\star \cdot S_a$	$\emptyset$
After-until	$Act^\star \cdot S_a$	S_b

(b) For behaviours.

Behaviour	ρ_b	α_f
Existence	ε	S_r
Existence at least k	$\sum_{0 \leq i < k}(\overline{\alpha_e \cup S_r}^\star \cdot S_r)^i$	S_r
Response	$\overline{\alpha_e}^\star \cdot S_q$	S_r
Chain response	See Appendix B	

Section 4. We express the non-existence of such a path, rather than expressing the equivalent notion that all complete paths satisfy the property, because we find the resulting formulae to be more intuitive. We first present a formula for $\mathcal{B}$-progress only. Subsequently, we give the formulae for weak fairness, weak hyperfairness and justness using a common structure all three share. Finally, we present the formulae for strong fairness and strong hyperfairness. In the justness and fairness formulae, $\mathcal{B}$-progress is also included: these assumptions eliminate unrealistic infinite paths, but we still need progress to discard unrealistic finite paths.

The proofs of all theorems in this section are included in Appendix D.

6.1 Progress

A formula for the non-existence of a violating path without progress is uninteresting. If progress is not assumed then all finite paths are complete, and therefore a path consisting of just ρ is a violating path whenever $\alpha_f \neq \emptyset$. The non-existence of a violating path would then be captured by $\neg\langle\rho\rangle tt$. This is why we include progress in all our formulae.

To represent progress, we must capture that as long as non-blocking actions are enabled, some transitions must still be executed. The following formula captures the non-existence of violating paths under $\mathcal{B}$-progress:

$$\neg\langle\rho\rangle\nu X.(\langle\alpha_e\rangle tt \vee [\overline{\mathcal{B}}]\mathit{ff} \vee \langle\overline{\alpha_f}\rangle X) \tag{1}$$

Intuitively, this formula says that there is no path that starts with a prefix matching ρ, after which infinitely often a transition can be taken that is not labelled with an action in α_f, or such transitions can be taken finitely often before a state is reached that is $\mathcal{B}$-locked or in which α_e is enabled. In the former case there is a $\mathcal{B}$-progressing path on which no actions in α_f occur after ρ. If a state in which α_e is enabled is reached, then it is guaranteed a violating and $\mathcal{B}$-progressing path exists: by arbitrarily extending the path as long as non-blocking actions are still enabled, a $\mathcal{B}$-progressing and violating path can be constructed.

▶ **Theorem 21.** *A state in an LTS satisfies Formula 1 if, and only if, it does not admit $\mathcal{B}$-progressing paths that are $(\rho, \alpha_f, \alpha_e)$-violating.*

Since representing a liveness pattern without progress leads to uninteresting formulae, it is unsurprising that previous translations of PSP to the μ-calculus have also implicitly included progress. For instance, the translations from [32] for the liveness patterns of PSP are very similar to Formula 1, albeit in positive form and without blocking actions.

6.2 Weak Fairness, Weak Hyperfairness and Justness

For weak fairness, weak hyperfairness and justness, we employ a trick inspired by the formula for justness presented in [7] (which was in turn inspired by [12]): we translate a requirement

on a full path into an invariant that can be evaluated within finitely many steps from every state of the path. We illustrate this using weak fairness.

On every suffix of a weakly fair path, every perpetually enabled non-blocking action occurs. To turn this into an invariant, we observe that we can evaluate a property on all suffixes of a path by evaluating it from every state of the path instead. Next we must determine, within finitely many steps, if an action is perpetually enabled on a possibly infinite path. We do this by observing that if an action is not perpetually enabled, it must become disabled within finitely many steps. An equivalent definition of WFA therefore is: a path π satisfies WFA if, and only if, for every state s in π, every action $a \in \overline{\mathcal{B}}$ that is enabled in s occurs or becomes disabled within finitely many steps on the suffix of π starting in s. This translation of WFA determines three things for every non-blocking action a. First, which actions may need to occur because of a; in the case of WFA this is a itself. Second, when those actions need to occur; for WFA this is when a is enabled. We refer to this as the action being "on". Finally, when those actions do not need to occur; for WFA this is when a becomes disabled. We refer to this as the action being "off". When an action that was previously on becomes off, or one of the required actions occurs, we say the action is "eliminated". By choosing different definitions for an action being on or off, and when an action is eliminated, we can also represent justness and weak hyperfairness in the same way.

We find that completeness criteria for which such a translation can be made can be represented using the same generalised formula. We will present this formula and how to instantiate it for WFA, WHFA and JA. However, we must first formalise what it means for a predicate on paths to be translatable to an invariant that can be evaluated within finitely many steps. We introduce the term *finitely realisable (path) predicates* for this purpose.

▶ **Definition 22.** *A path predicate P is* finitely realisable *if, and only if, there exist mappings ϕ_{on} and ϕ_{of} from non-blocking actions to closed modal μ-calculus formulae, and a mapping α_{el} from non-blocking actions to sets of actions, such that:*

1. *A path π satisfies predicate P if, and only if, all states s on π satisfy the following: for all $a \in \overline{\mathcal{B}}$, if s satisfies $\phi_{on}(a)$ then the suffix π' of π starting in s must contain an occurrence of some action in $\alpha_{el}(a)$ or a state that satisfies $\phi_{of}(a)$.*
2. *A state s is a $\mathcal{B}$-locked state if, and only if, $s \notin [\![\phi_{on}(a)]\!]$ for all $a \in \overline{\mathcal{B}}$.*
3. *For every state s and for all $a \in \overline{\mathcal{B}}$, $s \in [\![\phi_{on}(a)]\!]$ implies $s \notin [\![\phi_{of}(a)]\!]$.*
4. *For all states s and all $a \in \overline{\mathcal{B}}$ such that $s \in [\![\phi_{on}(a)]\!]$, if there exists a finite path π from s to a state s' such that there is no occurrence of an action in $\alpha_{el}(a)$ on π and there is no state on π that satisfies $\phi_{of}(a)$, then $s' \in [\![\phi_{on}(a)]\!]$.*

We refer to these four properties as the invariant property, *the* locking property, *the* exclusive property *and the* persistent property, *respectively.*

The general formula for finitely realisable predicates is as follows:

$$\neg\langle\rho\rangle\nu X.(\bigwedge_{a\in\overline{\mathcal{B}}} (\phi_{on}(a) \Rightarrow \langle\overline{\alpha_f}^\star\rangle(\langle\alpha_e\rangle tt \vee (\phi_{of}(a) \wedge X) \vee \langle\alpha_{el}(a) \setminus \alpha_f\rangle X))) \tag{2}$$

This formula has similarities to Formula 1, particularly how ρ and α_e are integrated. The important part is that after ρ, it must invariantly hold that all non-blocking actions for which $\phi_{on}(a)$ is satisfied are later eliminated. An action a is eliminated if, within finitely many steps, $\phi_{of}(a)$ is satisfied or an action in $\alpha_{el}(a)$ occurs. In both cases, the invariant must once again hold. After ρ, no actions in α_f may occur. The formula works correctly for finite paths as well as infinite ones: if it is possible to reach a $\mathcal{B}$-locked state after ρ without taking actions in α_f, then X is satisfied due to the locking property, and a violating path is found.

Formula 2 is a template formula in two ways: ρ, α_f and α_e determine what property is captured, and ϕ_{on}, ϕ_{of} and α_{el} determine the completeness criterion. In this paper, we only cover how to instantiate the formula for WFA, WHFA and JA, but it can also be used for other finitely realisable predicates. However, the correctness proof of the formula depends on the criterion being *feasible*. Feasibility on paths [3] is defined as follows.

▶ **Definition 23.** *A predicate on paths P is* feasible *if, and only if, for every LTS M, every finite path π in M can be extended to a path π' that satisfies P and is still a valid path in M.*

That WFA, WHFA and JA are feasible for finite LTSs is proven in Appendix C.

▶ **Theorem 24.** *For all feasible and finitely realisable path predicates P, it holds that an LTSC satisfies Formula 2 if, and only if, its initial state does not admit $\mathcal{B}$-progressing paths that satisfy P and are $(\rho, \alpha_f, \alpha_e)$-violating.*

By instantiating the theorem for each completeness criterion, we derive the following:

▶ **Corollary 25.** *A state in an LTS satisfies Formula 2 with $\phi_{on}(a) = \langle a \rangle tt$, $\phi_{of}(a) = [a]ff$ and $\alpha_{el}(a) = \{a\}$ for all $a \in \overline{\mathcal{B}}$ if, and only if, it does not admit $\mathcal{B}$-progressing paths that satisfy $\mathcal{B}$-weak fairness of actions and are $(\rho, \alpha_f, \alpha_e)$-violating.*

▶ **Corollary 26.** *A state in an LTS satisfies Formula 2 with $\phi_{on}(a) = \langle \overline{\mathcal{B}}^\star \cdot a \rangle tt$, $\phi_{of}(a) = [\overline{\mathcal{B}}^\star \cdot a]ff$ and $\alpha_{el}(a) = \{a\}$ for all $a \in \overline{\mathcal{B}}$ if, and only if, it does not admit $\mathcal{B}$-progressing paths that satisfy weak $\mathcal{B}$-hyperfairness of actions and are $(\rho, \alpha_f, \alpha_e)$-violating.*

▶ **Corollary 27.** *A state in an LTSC satisfies Formula 2 with $\phi_{on}(a) = \langle a \rangle tt$, $\phi_{of}(a) = ff$ and $\alpha_{el}(a) = \{b \in Act \mid a \not\rightarrowtail b\}$ for all $a \in \overline{\mathcal{B}}$ if, and only if, it does not admit $\mathcal{B}$-progressing paths that satisfy $\mathcal{B}$-justness of actions and are $(\rho, \alpha_f, \alpha_e)$-violating.*

6.3 Strong Fairness and Strong Hyperfairness

SFA is not finitely realisable because we cannot observe within finitely many steps whether an action is relentlessly enabled: even if we observe several times that it is disabled, it may still be infinitely often enabled along the whole path. Hence, we cannot use Formula 2.

Instead we observe that, on a path, actions that are not relentlessly enabled must eventually become perpetually disabled. If the path is strongly fair, then all relentlessly enabled non-blocking actions occur infinitely often. We can therefore say that a path is strongly fair if we can divide all non-blocking actions into two disjoint sets: those that occur infinitely often and those that eventually become perpetually disabled. This observation is also made in [37], where a μ-calculus formula for termination under strong fairness is given.

Using this idea, we give the following template formula for SFA:

$$\neg \langle \rho \cdot \overline{\alpha_f}^\star \rangle (\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}]ff \vee \bigvee_{\emptyset \neq F \subseteq \overline{\mathcal{B}}} \nu X.(\bigwedge_{a \in F} \mu W.((\bigwedge_{b \in \overline{\mathcal{B}} \setminus F} [b]ff) \wedge (\langle a \setminus \alpha_f \rangle X \vee \langle \overline{\alpha_f} \rangle W)))) \quad (3)$$

The use of negation, the exclusion of α_f, and ρ in the diamond operator at the start of this formula are the same as in Formula 1. We explain the start of the formula after addressing the part starting with $\bigvee_{\emptyset \neq F \subseteq \overline{\mathcal{B}}}$. Here, we use that on a strongly fair path, all non-blocking actions can be divided into those that occur infinitely often and those that become perpetually disabled. The disjunction over subsets considers all possible ways of selecting some non-empty subset F of $\overline{\mathcal{B}}$ that should occur infinitely often. The greatest fixpoint states that infinitely

often, all those actions must indeed occur within finitely many steps. Additionally, at no point may a non-blocking action not in F be enabled. We exclude $F = \emptyset$ because the logic of the greatest fixed point formula we give relies on there being at least one a in F. The special case that F is empty and therefore a $\mathcal{B}$-locked state should be reached, is instead covered by explicitly considering $[\overline{\mathcal{B}}]\mathit{ff}$ earlier in the formula. Returning to the start of the formula, we allow a finite α_f-free path before the greatest fixpoint is satisfied. The reason is that it may take several steps before all the non-blocking actions that are only finitely often enabled become perpetually disabled. Since we include a finite prefix already, we also add the cases that an action in α_e becomes enabled or that a $\mathcal{B}$-locked state is reached here, rather than deeper into the formula like in Formula 2.

▶ **Theorem 28.** *An LTS satisfies Formula 3 if, and only if, its initial state does not admit $\mathcal{B}$-progressing paths that satisfy $\mathcal{B}$-strong fairness of actions and are $(\rho, \alpha_f, \alpha_e)$-violating.*

Due to the quantification over subsets, the formula is exponential in the number of actions in $\overline{\mathcal{B}}$. Beyond small models, it is therefore not practical. However, it can serve as a basis for future work. For instance, if fairness is applied to sets of actions rather than individual actions, the formula is exponential in the number of sets instead, which may be smaller depending on how the sets are formed [36].

We can adapt the formula for strong fairness to a formula for strong hyperfairness, by replacing perpetual disabledness of non-blocking actions not in F with perpetual unreachability.

$$\neg\langle\rho\cdot\overline{\alpha_f}^\star\rangle(\langle\alpha_e\rangle tt\vee[\overline{\mathcal{B}}]\mathit{ff}\vee\bigvee_{\emptyset\neq F\subseteq\overline{\mathcal{B}}}\nu X.(\bigwedge_{a\in F}\mu W.((\bigwedge_{b\in\overline{\mathcal{B}}\setminus F}[\overline{\mathcal{B}}^\star\cdot b]\mathit{ff})\wedge(\langle a\setminus\alpha_f\rangle X\vee\langle\overline{\alpha_f}\rangle W)))) \quad (4)$$

▶ **Theorem 29.** *An LTS satisfies Formula 4 if, and only if, its initial state does not admit a $\mathcal{B}$-progressing path that satisfies strong $\mathcal{B}$-hyperfairness of actions and is $(\rho, \alpha_f, \alpha_e)$-violating.*

Since we are not aware of other completeness criteria that fit the same structure, we do not provide a generalised formula here like we did with Formula 2, although we do prove a more general theorem in Appendix D.4.

7   Application Example

We here give an example of an application of the template formulae. In [26], several mutual exclusion algorithms are analysed using the mCRL2 toolset. Their analysis of Dekker's algorithm [15] presents the following modal μ-calculus formula for starvation freedom of processes with id's 0 and 1. For clarity, the notation has been adjusted to match the previous sections and action names have been simplified.

$$[Act^\star]\bigwedge_{i\in\{0,1\}}[\{wish\_flag(i,b)\mid b\in\mathbb{B}\}]\mu X.([\overline{enter(i)}]X\wedge\langle Act\rangle tt) \quad (5)$$

Starvation freedom is a *global response* property. In this case, the starvation freedom of a process i is represented as an instantiation of the pattern with $S_q = \{wish\_flag(i,b)\mid b\in\mathbb{B}\}$ and $S_r = \{enter(i)\}$. Indeed, the above formula is equivalent to:

$$\bigwedge_{i\in\{0,1\}}\neg\langle Act^\star\cdot\{wish\_flag(i,b)\mid b\in\mathbb{B}\}\rangle\nu X.(\langle\emptyset\rangle tt\vee[Act]\mathit{ff}\vee\langle\overline{enter(i)}\rangle X) \quad (6)$$

Observe that, when taking $\mathcal{B} = \emptyset$, the above matches a conjunction of two instances of Formula 1, taking ρ, α_e and α_f as suggested in Table 1 for *global response*. Thus, this formula captures starvation freedom under $\emptyset$-progress. In [26], it is reported that mCRL2 finds a violating path for this formula; a path which the authors note is unfair. The exact fairness assumption considered is not made concrete. As an ad-hoc solution, the modal μ-calculus formula is adjusted to specifically ignore that counterexample. Subsequently, mCRL2 finds another counterexample, which the authors again claim is unfair. Instead of creating yet another formula, they move on to Peterson's algorithm, which is deemed easier to analyse. Using our template formulae, we can easily produce a formula for starvation freedom under several different completeness criteria. We give the formula for $\emptyset$-WFA, as an example.

$$\bigwedge_{i \in \{0,1\}} \neg \langle Act^\star \cdot \{ wish\_flag(i, b) \mid b \in \mathbb{B} \} \rangle$$
$$\nu X.(\bigwedge_{a \in Act} (\langle a \rangle tt \Rightarrow \langle \overline{enter(i)}^\star \rangle (\langle \emptyset \rangle tt \vee ([a]ff \wedge X) \vee \langle a \setminus enter(i) \rangle X))) \quad (7)$$

We check this formula on the model from [26] using mCRL2. Since mCRL2 only supports quantification over data parameters and not over actions, the conjunction over Act must be written out explicitly. The tool reports that the formula is violated. Examining the counterexample reveals this is because actions in the model do not show which process performs the action. Therefore, process i reading value v from a register r is labelled with the same action as process j reading v from r. We add the responsible process to each action label, and also define $\mathcal{B} = \{ wish\_flag(i, i, b) \mid i \in \{0, 1\}, b \in \mathbb{B} \}$, to capture that processes are allowed to remain in their non-critical section indefinitely. This was not considered in Formula 5, but it is part of the mutual exclusion problem [16, 23]. The tool reports that the modified formula is satisfied. We can therefore conclude that Dekker's algorithm satisfies starvation freedom when assuming weak fairness of actions, as long as it is taken into account for each action which process is responsible for it.

Our other formulae can be used in similar ways. An example of how to use the justness formula in mCRL2, including a method for encoding the concurrency relation, is given in [7].

8 Discussion

In this section, we briefly reflect on the coverage of the properties we consider, and our choice in focusing on the modal μ-calculus.

Firstly, we have exclusively addressed liveness properties in this paper thus far. As indicated previously, the problem we are considering primarily crops up for these properties. This is because, as pointed out in [23], when a completeness criterion is feasible, assuming the criterion holds true or not has no impact on whether a safety property is satisfied or not. The reason is that for safety properties on paths, any path that violates the property must contain a finite prefix such that any extension of that prefix also violates the property [1]. Therefore, if a completeness criterion is feasible, then whenever a model contains incomplete paths that violate a safety property it also contains complete paths that violate the property. All completeness criteria discussed in Section 4 are feasible with respect to finite LTSs, and hence we do not need to consider patterns that capture safety properties. For modal μ-calculus formulae for the safety properties of PSP, without integrated completeness criteria, we refer to [32] and [35]. For properties that are a combination of safety and liveness, the components can be turned into separate formulae and checked separately.

Readers may also wonder about alternative methods of representing properties under completeness criteria, such as using LTL. As indicated in Section 3, there are many contexts where we also want to consider non-linear properties, and hence the modal μ-calculus is preferred. Automatic translations from LTL to the modal μ-calculus exist, but can be exponential in complexity [14] and it is unclear at this time if this blow-up is avoided in this case. Anecdotal evidence [34] suggests this is not the case for existing translations. In [23] several completeness criteria are represented in LTL, but it is noted that this translation requires introducing new atomic propositions which hides the complexity of this translation. The representation of hyperfairness in particular may be expensive, since atomic propositions for all reachable actions are required. It is also unclear how to combine LTL-based translations effectively with symbolic model checking approaches. For these reasons, a direct representation in the modal μ-calculus is preferable.

9   Conclusion

In this paper, we have presented formulae for liveness properties under several completeness criteria. As part of this, we defined a property template that generalises the liveness properties of PSP, which has been estimated to cover a majority of properties found in the literature [17]. The completeness criteria covered are progress, justness, weak fairness, strong fairness, and hyperfairness, all defined with respect to actions and parameterised with a set of blocking actions. The formulae have all been manually proven to be correct.

For future work, one goal is to formalise the proofs in the appendices using a proof assistant. Another avenue for future work is extending our formulae to cover a wider range of completeness criteria and properties. We suggest some potential extensions here.

One of our contributions is the identification of a shared common structure underlying justness, weak fairness and weak hyperfairness: they are finitely realisable path predicates. Our formula for such predicates can be adapted to arbitrary feasible finitely realisable path predicates. While we do not have such a generic formula for other completeness criteria, our characterisation of $(\rho, \alpha_f, \alpha_e)$-violating paths can be used as a basis to express the non-existence of complete paths violating many common properties for different notions of completeness as well, as we demonstrate with strong fairness and strong hyperfairness. We are especially interested in extending our formulae to allow fairness over sets of actions, rather than individual actions, similar to the task-based definitions from [25].

In terms of properties, we can look at proposed extensions of PSP, such as those suggested in [13]. There is also the *constrained chain* behaviour, which is a modification of precedence chain and response chain given in [17]. There are extensions of PSP to real-time [5, 29] and probabilistic [28] contexts as well. Finally, in [6] the formula from [7] that formed the basis of Formula 2 is extended to also include state information.

There are therefore many potentially useful extensions of the formulae presented in this paper. However, the presented template formulae already cover many completeness criteria and liveness properties, making them useful for model checking in practice.

References

1   Mack W. Alford, Leslie Lamport, and Geoff P. Mullery. Basic concepts. In Mack W. Alford, Jean-Pierre Ansart, Günter Hommel, Leslie Lamport, Barbara Liskov, Geoff P. Mullery, and Fred B. Schneider, editors, *Distributed Systems: Methods and Tools for Specification, An Advanced Course, April 3-12, 1984 and April 16-25, 1985, Munich, Germany*, volume 190 of

Lecture Notes in Computer Science, pages 7–43. Springer, 1984. `doi:10.1007/3-540-15216-4\_12`.

2 Bowen Alpern and Fred B. Schneider. Defining liveness. *Inf. Process. Lett.*, 21(4):181–185, 1985. `doi:10.1016/0020-0190(85)90056-0`.

3 Krzysztof R. Apt, Nissim Francez, and Shmuel Katz. Appraising fairness in languages for distributed programming. *Distributed Comput.*, 2(4):226–241, 1988. `doi:10.1007/BF01872848`.

4 Paul C. Attie, Nissim Francez, and Orna Grumberg. Fairness and hyperfairness in multiparty interactions. In Frances E. Allen, editor, *Conference Record of the Seventeenth Annual ACM Symposium on Principles of Programming Languages, San Francisco, California, USA, January 1990*, pages 292–305. ACM Press, 1990. `doi:10.1145/96709.96739`.

5 Pierfrancesco Bellini, Paolo Nesi, and Davide Rogai. Expressing and organizing real-time specification patterns via temporal logics. *J. Syst. Softw.*, 82(2):183–196, 2009. `doi:10.1016/j.jss.2008.06.041`.

6 Mark S. Bouwman. *Supporting Railway Standardisation with Formal Verification*. Phd Thesis 1 (Research TU/e / Graduation TU/e), Mathematics and Computer Science, Eindhoven University of Technology, 2023. `https://pure.tue.nl/ws/portalfiles/portal/307965423/20231023_Bouwman_hf.pdf`.

7 Mark S. Bouwman, Bas Luttik, and Tim A. C. Willemse. Off-the-shelf automated analysis of liveness properties for just paths. *Acta Informatica*, 57(3-5):551–590, 2020. `doi:10.1007/s00236-020-00371-w`.

8 Julian C. Bradfield and Colin Stirling. Modal logics and mu-calculi: an introduction. In Jan A. Bergstra, Alban Ponse, and Scott A. Smolka, editors, *Handbook of Process Algebra*, pages 293–330. Elsevier Science, 2001. `doi:10.1016/b978-044482830-9/50022-9`.

9 Julian C. Bradfield and Colin Stirling. Modal mu-calculi. In Patrick Blackburn, Johan Van Benthem, and Frank Wolter, editors, *Handbook of Modal Logic*, volume 3 of *Studies in logic and practical reasoning*, pages 721–756. Elsevier, 2007. `doi:10.1016/s1570-2464(07)80015-2`.

10 Julian C. Bradfield and Igor Walukiewicz. The mu-calculus and model checking. In Edmund M. Clarke, Thomas A. Henzinger, Helmut Veith, and Roderick Bloem, editors, *Handbook of Model Checking*, pages 871–919. Springer, 2018. `doi:10.1007/978-3-319-10575-8\_26`.

11 Olav Bunte, Jan Friso Groote, Jeroen J. A. Keiren, Maurice Laveaux, Thomas Neele, Erik P. de Vink, Wieger Wesselink, Anton Wijs, and Tim A. C. Willemse. The mCRL2 toolset for analysing concurrent systems. In Tomáš Vojnar and Lijun Zhang, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 25th International Conference, TACAS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings, Part II*, volume 11428 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2019. `doi:10.1007/978-3-030-17465-1\_2`.

12 Edmund M. Clarke, Orna Grumberg, Kenneth L. McMillan, and Xudong Zhao. Efficient generation of counterexamples and witnesses in symbolic model checking. In Bryan Preas, editor, *Proceedings of the 32st Conference on Design Automation, San Francisco, California, USA, Moscone Center, June 12-16, 1995*, pages 427–432. ACM Press, 1995. `doi:10.1145/217474.217565`.

13 Rachel L. Cobleigh, George S. Avrunin, and Lori A. Clarke. User guidance for creating precise and accessible property specifications. In Michal Young and Premkumar T. Devanbu, editors, *Proceedings of the 14th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE 2006, Portland, Oregon, USA, November 5-11, 2006*, pages 208–218. ACM, 2006. `doi:10.1145/1181775.1181801`.

14 Sjoerd Cranen, Jan Friso Groote, and Michel A. Reniers. A linear translation from CTL* to the first-order modal μ-calculus. *Theor. Comput. Sci.*, 412(28):3129–3139, 2011. `doi:10.1016/j.tcs.2011.02.034`.

15   Edsger W Dijkstra. Over de sequentialiteit van procesbeschrijvingen (EWD-35). EW dijkstra archive. *Center for American History, University of Texas at Austin*, 1962. URL: `https://www.cs.utexas.edu/~EWD/ewd00xx/EWD35.PDF`.

16   Edsger W. Dijkstra. Solution of a problem in concurrent programming control. *Commun. ACM*, 8(9):569, 1965. `doi:10.1145/365559.365617`.

17   Matthew B. Dwyer, George S. Avrunin, and James C. Corbett. Patterns in property specifications for finite-state verification. In Barry W. Boehm, David Garlan, and Jeff Kramer, editors, *Proceedings of the 1999 International Conference on Software Engineering, ICSE' 99, Los Angeles, CA, USA, May 16-22, 1999*, pages 411–420. ACM, 1999. `doi:10.1145/302405.302672`.

18   E. Allen Emerson and Edmund M. Clarke. Using branching time temporal logic to synthesize synchronization skeletons. *Sci. Comput. Program.*, 2(3):241–266, 1982. `doi:10.1016/0167-6423(83)90017-5`.

19   Michael J. Fischer and Richard E. Ladner. Propositional dynamic logic of regular programs. *J. Comput. Syst. Sci.*, 18(2):194–211, 1979. `doi:10.1016/0022-0000(79)90046-1`.

20   Hubert Garavel, Frédéric Lang, Radu Mateescu, and Wendelin Serwe. CADP 2011: a toolbox for the construction and analysis of distributed processes. *Int. J. Softw. Tools Technol. Transf.*, 15(2):89–107, 2013. `doi:10.1007/s10009-012-0244-z`.

21   Rob J. van Glabbeek. Justness - A completeness criterion for capturing liveness properties (extended abstract). In Mikołaj ojańczyk and Alex Simpson, editors, *Foundations of Software Science and Computation Structures - 22nd International Conference, FOSSACS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings*, volume 11425 of *Lecture Notes in Computer Science*, pages 505–522. Springer, 2019. `doi:10.1007/978-3-030-17127-8\_29`.

22   Rob J. van Glabbeek. Reactive temporal logic. In Ornela Dardha and Jurriaan Rot, editors, *Proceedings Combined 27th International Workshop on Expressiveness in Concurrency and 17th Workshop on Structural Operational Semantics, EXPRESS/SOS 2020, and 17th Workshop on Structural Operational SemanticsOnline, 31 August 2020*, volume 322 of *EPTCS*, pages 51–68. Open Publishing Association, 2020. `doi:10.4204/EPTCS.322.6`.

23   Rob J. van Glabbeek. Modelling mutual exclusion in a process algebra with time-outs. *Inf. Comput.*, 294:105079, 2023. `doi:10.1016/j.ic.2023.105079`.

24   Rob J. van Glabbeek and Peter Höfner. CCS: It's not fair! fair schedulers cannot be implemented in CCS-like languages even under progress and certain fairness assumptions. *Acta Informatica*, 52(2-3):175–205, 2015. `doi:10.1007/s00236-015-0221-6`.

25   Rob J. van Glabbeek and Peter Höfner. Progress, Justness, and Fairness. *ACM Comput. Surv.*, 52(4):69:1–69:38, 2019. `doi:10.1145/3329125`.

26   Jan Friso Groote and Jeroen J. A. Keiren. Tutorial: designing distributed software in mCRL2. In Kirstin Peters and Tim A. C. Willemse, editors, *Formal Techniques for Distributed Objects, Components, and Systems - 41st IFIP WG 6.1 International Conference, FORTE 2021, Held as Part of the 16th International Federated Conference on Distributed Computing Techniques, DisCoTec 2021, Valletta, Malta, June 14-18, 2021, Proceedings*, volume 12719 of *Lecture Notes in Computer Science*, pages 226–243. Springer, 2021. `doi:10.1007/978-3-030-78089-0\_15`.

27   Jan Friso Groote and Mohammad Reza Mousavi. *Modeling and Analysis of Communicating Systems*. MIT Press, 08 2014. URL: `https://mitpress.mit.edu/books/modeling-and-analysis-communicating-systems`.

28   Lars Grunske. Specification patterns for probabilistic quality properties. In Wilhelm Schäfer, Matthew B. Dwyer, and Volker Gruhn, editors, *30th International Conference on Software Engineering (ICSE 2008), Leipzig, Germany, May 10-18, 2008*, pages 31–40. ACM, 2008. `doi:10.1145/1368088.1368094`.

29   Sascha Konrad and Betty H. C. Cheng. Real-time specification patterns. In Gruia-Catalin Roman, William G. Griswold, and Bashar Nuseibeh, editors, *27th International Conference on*

Software Engineering (ICSE 2005), 15-21 May 2005, St. Louis, Missouri, USA, pages 372–381. ACM, 2005. `doi:10.1145/1062455.1062526`.

30 Dexter Kozen. Results on the propositional μ-calculus. *Theor. Comput. Sci.*, 27(3):333–354, 1983. `doi:10.1016/0304-3975(82)90125-6`.

31 Leslie Lamport. Fairness and hyperfairness. *Distributed Comput.*, 13(4):239–245, 2000. `doi:10.1007/PL00008921`.

32 Radu Mateescu. Property Pattern Mappings for RAFMC, 2019. Available at: `https://cadp.inria.fr/resources/evaluator/rafmc.html` (Accessed: 26 January 2024).

33 Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, pages 46–57. IEEE Computer Society, 1977. `doi:10.1109/SFCS.1977.32`.

34 Jaco van de Pol and Michael Weber. A multi-core solver for parity games. *Electronic Notes in Theoretical Computer Science*, 220(2):19–34, 2008. Proceedings of the 7th International Workshop on Parallel and Distributed Methods in verifiCation (PDMC 2008). `doi:10.1016/j.entcs.2008.11.011`.

35 Daniela Remenska. *Bringing Model Checking Closer To Practical Software Engineering*. PhD thesis, Vrije U., Amsterdam, 2016. PhD Thesis, available at: `https://hdl.handle.net/1871/53958`.

36 Myrthe S. C. Spronck. Fairness assumptions in the modal μ-calculus, 2023. Master's thesis, Eindhoven University of Technology, available at `https://research.tue.nl/en/studentTheses/fairness-assumptions-in-the-modal-%C2%B5-calculus`.

37 Frank A. Stomp, Willem-Paul de Roever, and Rob T. Gerth. The μ-calculus as an assertion-language for fairness arguments. *Inf. Comput.*, 82(3):278–322, 1989. `doi:10.1016/0890-5401(89)90004-7`.

A   Property Patterns

Here we recall the behaviours and scopes presented in [17]. The original presentation is not restricted to a particular logic and the patterns allow behaviour and scopes to be defined based on both states and actions. We give the definitions specifically with respect to occurrences of actions, since those are the properties we consider in this paper. We use S_a ("after"), S_b ("before"), S_q ("query") and S_r ("required"/"response") as placeholder names for property-specific sets of actions. We use k for an arbitrary natural number.

The following behaviours are given:

- **Absence**: no action in S_r may occur.
- **Existence**: some action in S_r must occur.
  - **Existence at least/at most/exactly**: there must be at least/at most/exactly k occurrences of actions in S_r. The *existence* pattern is an instantiation of *existence at least* with $k = 1$.
- **Universality**: only actions in S_r occur.
- **Precedence**: an occurrence of an action in S_r must always be preceded by an occurrence of an action in S_q.
  - **Chain precedence**: if actions from the sets S_{r_0}, S_{r_1}, …, S_{r_n} occur in that order (potentially with other actions in-between), then they must have been preceded by occurrences of actions from the sets S_{q_0}, S_{q_1}, …, S_{q_m}, in that order.
- **Response**: an occurrence of an action in S_q must be followed by the occurrence of an action from S_r.
  - **Chain response**: if actions from the sets S_{q_0}, S_{q_1}, …, S_{q_n} occur in that order (potentially with other actions in between), then they must be followed by occurrences of actions from the sets S_{r_0}, S_{r_1}, …, S_{r_m}, in that order.

All of the behaviours only need to hold within the chosen scope. The following scopes are given:

- **Global**: the full path.
- **Before**: the prefix of the path before the first occurrence of an action in S_b. If no action in S_b occurs on the path, then the behaviour does not need to be satisfied anywhere.
  - **Until**: same as *before*, except that if no action in S_b occurs, then the behaviour needs to hold on the full path.
- **After**: the suffix of the path after the first occurrence of an action in S_a. If no action in S_a occurs on the path, then the behaviour does not need to be satisfied anywhere.
- **Between**: every subpath of the path that starts after an occurrence of an action in S_a and ends before the first following occurrence of an action in S_b. If there is an occurrence of an action in S_a that is not eventually followed by an action in S_b, the behaviour does not need to be satisfied after that S_a. This combines *after* and *before*, but unlike the default *after* scope considers any occurrence of S_a, not merely the first.
  - **After-until**: same as *between*, except that if there is an occurrence of an action in S_a that is not eventually followed by an action in S_b, the behaviour still needs to be satisfied after that occurrence of S_a. This combines *after* and *until*.

The *until* scope does not appear in [17], but *after-until* does. We include the *until* scope from [35], there called *before-variant*, because it can be seen as a simpler form of *after-until*.

We only consider liveness properties, so we must ask which combinations of behaviour and scope result in liveness properties. To make this judgement, we need a formal definition of what makes a property a safety or liveness property. For the purposes of this paper, since

all the properties we consider are defined on occurrences of actions, we can use the following definition of a property:

▶ **Definition 30.** *A* property *is a set of sequences of actions. A path π satisfies* the property *if its sequence of actions is in the set, otherwise it* violates *the property.*

We adapt the formal definitions of safety and liveness properties from [1] and [2] respectively to this definition of properties.

▶ **Definition 31.** *A property P is a* safety *property if, and only if, every infinite sequence of actions not in P has a finite prefix that is not in P.*

The consequence of this is that an infinite path that violates a safety property always has a finite prefix that violates it as well.

▶ **Definition 32.** *A property P is a* liveness *property if, and only if, for every finite sequence of actions f there exists some infinite sequence of actions f' such that ff' is in P.*

In terms of paths, this means that for every finite path π that violates a liveness property, there exists an infinite path π' of which π is a prefix that satisfies the property. Note that it is not required for an LTS that admits π to also admit π', only that such an extension could be made. We now discuss which patterns form liveness properties.

First, we note that the *before* and *between* scopes will turn every behaviour into a safety property: whenever an action in S_b occurs the behaviour should be satisfied before that occurrence, hence every path that violates the property will have a finite prefix, ending with the first occurrence of an action in S_b, that also violates the property. The *global*, *until*, *after* and *after-until* scopes remain relevant. Under these four scopes, the behaviours *absence*, *existence at most*, *universality*, *precedence* and *chain precedence* will always result in safety properties as well. For each of these behaviours, some actions may not occur under certain circumstances (be it at all, after there have already been a number of occurrence of those actions, or when some other actions have not yet occurred), therefore every violating path has a finite prefix, ending with the occurrence of such an action, that also violates the path. This leaves us with *existence*, *existence at least*, *existence exactly*, *response* and *chain response*.

We further drop *existence exactly* since it is merely a conjunction of *existence at least* and *existence at most*. Both parts of the pattern can be expressed separately, so a separate formula for *existence exactly* is superfluous.

We could apply a similar argument to the *until* scope: it is merely a combination of *global* and *before*. Saying that an action in S_r has to occur until S_b (*existence until*), for instance, is the same as saying that an action in S_r has to occur at all (*global existence*) and that if there is an occurrence of S_b, there must be an occurrence of S_r before it (*existence before*). This extends to *after-until* as well, although it requires a bit more care than simply combining *after* and *between*, since *after* always applies to the first occurrence of an action in S_a, whereas *after-until* refers to every occurrence. This could be achieved with minor modifications to the patterns. However, it turns out we can relatively easily incorporate the *until* and *after-until* scopes into our formulae, so we include them for convenience.

B   Representing *Chain Response*

We here illustrate how the *chain response* behaviour can be represented using our template formulae by combining several *response* formulae.

Consider, for example, sequences of two sets each: if an occurrence of S_{q_0} is eventually followed by an occurrence of S_{q_1}, then there must subsequently be an occurrence of S_{r_0}

followed by S_{r_1}. There are two possible violating paths here: either S_{q_0}, S_{q_1} is not followed by S_{r_0}, or $S_{q_0}, S_{q_1}, S_{r_0}$ is not followed by S_{r_1}. These two violations cannot be slotted directly into the form of a single $(\rho, \alpha_f, \alpha_e)$-violating path. Instead, we have two different $(\rho, \alpha_f, \alpha_e)$-violating paths. In general, if we have chain response with S_{q_0} to S_{q_n} and S_{r_0} to S_{r_m} then we get m different violating paths. Specifically, for all $0 \le i \le m$, we get a violating path that consists of a sequence $S_{q_0}, \ldots, S_{q_n}, S_{r_0}, \ldots, S_{r_{i-1}}$ that may not be followed by an occurrence of S_{r_i}. For convenience, we write such violating paths as a sequence $S_0, S_1, \ldots, S_n$ that may not be followed by S_{n+1}.

Such violating paths can be expressed through $\rho_b = \overline{\alpha_e}^\star \cdot S_0 \cdot \overline{\alpha_e \cup S_1}^\star \cdot S_1 \cdot \overline{\alpha_e \cup S_2}^\star \cdot S_2 \ldots \cdot \overline{\alpha_e \cup S_n}^\star \cdot S_n$, and $\alpha_f = S_{n+1}$. Each violating path must be given its own formula, where Table 1a is still used for the scope, and all resulting formulae placed in conjunction. This way *chain-response* can be represented.

▶ **Example 33.** Say we want to express *chain response* with the scope *after-until* under WFA, and we take the chain that an occurrence of an action in S_{q_0}, if followed by an action in S_{q_1}, needs to be followed by an action in S_{r_0} and then by an action in S_{r_1}. The violating paths are occurrences of an actions in S_{q_0}, S_{q_1} not followed by an action S_{r_0}, and occurrences of actions in $S_{q_0}, S_{q_1}, S_{r_0}$ with no subsequent occurrence of an action in S_{r_1}. Both must be after the first occurrence of an action in S_a, and before the next occurrence of an action in S_b. The formula we need is then:

$$\neg(\langle Act^\star \cdot S_a \cdot \overline{S_b}^\star \cdot S_{q_0} \cdot \overline{S_b \cup S_{q_1}}^\star \cdot S_{q_1} \rangle$$
$$\nu X.(\bigwedge_{a \in \overline{\mathcal{B}}} (\langle a \rangle tt \Rightarrow \langle \overline{S_{r_0}}^\star \rangle (\langle S_b \rangle tt \vee ([a]ff \wedge X) \vee \langle a \setminus S_{r_0} \rangle X))))$$
$$\wedge$$
$$\neg(\langle Act^\star \cdot S_a \cdot \overline{S_b}^\star \cdot S_{q_0} \cdot \overline{S_b \cup S_{q_1}}^\star \cdot S_{q_1} \cdot \overline{S_b \cup S_{r_0}}^\star \cdot S_{r_0} \rangle$$
$$\nu X.(\bigwedge_{a \in \overline{\mathcal{B}}} (\langle a \rangle tt \Rightarrow \langle \overline{S_{r_1}}^\star \rangle (\langle S_b \rangle tt \vee ([a]ff \wedge X) \vee \langle a \setminus S_{r_1} \rangle X))))$$

C   Proofs of Feasibility

In Section 6.2 we claimed WFA, WHFA and JA are feasible with respect to finite LTSs. In the proof of the SFA and SHFA formulae, we will need feasibility of SFA and SHFA as well. In this appendix, we give those proofs.

All our proofs assume a fixed LTSC $M = (\mathcal{S}, s_{init}, Act, Trans, \smile\!\!\bullet)$, although the $\smile\!\!\bullet$ is only relevant for JA. We also refer to an arbitrary environment e, and set of blocking actions $\mathcal{B} \subseteq Act$. When we refer to an arbitrary state or transition in a path in our proofs, it should be understood that we are referring to specific occurrences of those states and transitions unless explicitly stated otherwise.

Recall Definition 23:

▶ **Definition 23.** *A predicate on paths P is* feasible *if, and only if, for every LTS M, every finite path π in M can be extended to a path π' that satisfies P and is still a valid path in M.*

▶ **Proposition 34.** *$\mathcal{B}$-weak fairness of actions is feasible.*

Proof. It is proven in [25, Theorem 6.1] that if only countably many actions are enabled in each state of a transition system, then weak fairness of actions with $\mathcal{B} = \emptyset$ is feasible. We

have assumed a finite set Act, hence this theorem applies in our case. This means that every finite path can be extended to a path that is $\emptyset$-weakly fair. A path that satisfies $\emptyset$-WFA also satisfies $\mathcal{B}$-WFA for arbitrary $\mathcal{B}$, since $\emptyset$-WFA requires all actions in Act to occur in suffixes that they are perpetually enabled in, and Act is a superset of $\overline{\mathcal{B}}$. We conclude that $\mathcal{B}$-weak fairness of actions is feasible.   ◀

▶ **Proposition 35.** *$\mathcal{B}$-strong fairness of actions is feasible.*

Proof. It is proven in [25, Theorem 6.1] that if only countably many actions are enabled in each state of a transition system, then strong fairness of actions with $\mathcal{B} = \emptyset$ is feasible. We have assumed a finite set Act, hence this theorem applies. Similarly to WFA, as argued in Proposition 34, $\emptyset$-SFA implies $\mathcal{B}$-SFA for arbitrary $\mathcal{B}$ because $\mathcal{B}$-SFA requires only actions in $\overline{\mathcal{B}}$ to occur when they are relentlessly enabled. We conclude $\mathcal{B}$-strong fairness of actions is feasible.   ◀

For the two forms of hyperfairness, we first prove a supporting lemma.

▶ **Lemma 36.** *Every finite path π can be extended to a path π' that satisfies weak $\mathcal{B}$-hyperfairness of actions, such that all occurrences of blocking actions in π' are part of π.*

Proof. Let π be an arbitrary finite path. We prove that π can be extended to path π' that satisfies weak $\mathcal{B}$-hyperfairness of actions, such that there are no occurrences of blocking actions in the extension. We do this through construction of the path π'. We will construct π' in steps. Let π_i with $i \geq 0$ be the path constructed in the i'th iteration, with $\pi_0 = \pi$. Let s_i be the last state of π_i. For this construction, we use a queue Q containing non-blocking actions. At the start of the construction, Q is initialised with exactly one copy all non-blocking actions $\mathcal{B}$-reachable from s_0, the final state of π, in some arbitrary order. The construction has the following invariants: Q contains exactly one copy of every non-blocking action $\mathcal{B}$-reachable in the final state of the path constructed so far. It may contain zero or one copies of non-blocking actions not $\mathcal{B}$-reachable from this state. It contains no blocking actions. Additionally, the only occurrences of blocking actions in the path constructed thus far are in π.

At each step $i > 0$, we do the following: first, we determine if Q is empty or not. If it is empty, we take $\pi_i = \pi_{i-1}$ and the construction terminates. If Q is not empty, we pop the head a from Q. If a is not $\mathcal{B}$-reachable from s_{i-1}, then we let $\pi_i = \pi_{i-1}$ and go to step $i + 1$. The invariants are maintained because we only removed a from Q, and a was not $\mathcal{B}$-reachable from $s_{i-1} = s_i$. If a is $\mathcal{B}$-reachable from s_{i-1}, then there exists some path π'_i consisting of only non-blocking actions starting in s_{i-1} and ending in a state s'_{i-1} such that some transition t_a with $act(t_a) = a$ is enabled in s'_{i-1}. Let $\pi_i = \pi_{i-1} \cdot \pi'_{i-1} t_a \, trgt(t_a)$ and append a back to the end of Q. Then continue to step $i + 1$. The invariants are maintained in this case as well. This is because every action $\mathcal{B}$-reachable in s_i must also have been $\mathcal{B}$-reachable from s_{i-1}, since s_i is $\mathcal{B}$-reachable from s_{i-1}. Since Q at the start of this step contains the same actions as at the end of this step, and by the invariant it contained all actions at the start that are $\mathcal{B}$-reachable from s_{i-1}, it also contains all actions that are $\mathcal{B}$-reachable from s_i at the end of the step. Finally, the segment we added did not contain any blocking actions, and a itself is non-blocking because all actions in Q are non-blocking.

There are two potential outcomes to this construction: either Q becomes empty and the construction terminates, or Q never becomes empty and the construction continues infinitely. We prove that in either case, the path π' that is ultimately constructed is weakly $\mathcal{B}$-hyperfair

of actions and does not contain occurrences of blocking actions beyond those already present in π.

- If Q becomes empty and the construction terminates, then the final path π' is π_i for the i on which Q was determined to be empty. The final state of π', s' is then a state in which no non-blocking actions are $\mathcal{B}$-reachable. Hence, there are no non-blocking actions perpetually $\mathcal{B}$-reachable on any suffix of π' and so π' is trivially $\mathcal{B}$-WHFA.

- If Q never becomes empty then the construction continues forever. Let π' be the infinite path π_∞. Let π'' be an arbitrary suffix of π', and let a be an arbitrary action in $\overline{\mathcal{B}}$ that is perpetually $\mathcal{B}$-reachable on π''. We prove a occurs in π''. Consider that if a is perpetually $\mathcal{B}$-reachable on π'', then it is $\mathcal{B}$-reachable in every state of π''. Consider also that, since π'' is a suffix of the infinite path π', π'' is also infinite. In our construction, we add only a finite number of steps to the path in every iteration. Therefore, π'' was created as a part of π' over infinitely many iterations. Since a is enabled in every state of π'', by the invariants a must be in Q at the start of all iterations of the construction that contributed to π'', with possible exception of the first. Since Q is a queue and *Act* is finite, a will be at the head of the queue during the construction of π'' infinitely many times. Whenever a was at the head of the queue during the construction, a finite number of steps were added to the path that ended with a transion labelled with a. Hence, a occurs in π'', and so π' is $\mathcal{B}$-WHFA.

In both cases, that no new occurrences of blocking actions are added to the path comes directly from the invariants. ◀

▶ **Proposition 37.** *Weak $\mathcal{B}$-hyperfairness of actions is feasible.*

Proof. This follows from Lemma 36, which is a stronger property. ◀

▶ **Proposition 38.** *Strong $\mathcal{B}$-hyperfairness of actions in feasible.*

Proof. We prove that every finite path π can be extended to a path π' that satisfies strong $\mathcal{B}$-hyperfairness of actions. Let π be an arbitrary finite path, then by Lemma 36 we know that there exists a path π' that extends π and satisfies weak $\mathcal{B}$-hyperfairness of actions, and has no occurrences of blocking actions save those already present in π. We will use π' to witness that there exists a strongly $\mathcal{B}$-hyperfair extension of π, by proving π' satisfies strong $\mathcal{B}$-hyperfairness as well as weak $\mathcal{B}$-hyperfairness.

Towards a contradiction, assume that π' does not satisfy strong $\mathcal{B}$-hyperfairness of actions. Then π' must have a suffix π'' such that there is an action $a \in \overline{\mathcal{B}}$ that is relentlessly $\mathcal{B}$-reachable in π'' and yet does not occur in π''. If a is relentlessly $\mathcal{B}$-reachable in π'', it is also relentlessly $\mathcal{B}$-reachable in every suffix of π''. Let π''' be a suffix of π'' such that π''' does not contain any occurrences of blocking actions. That such a suffix exists follows from π' only having occurrences of blocking actions in the finite prefix π. We now have the path π''' on which a is relentlessly $\mathcal{B}$-reachable and that does not contain occurrences of blocking actions. Let s be an arbitrary state on π'''. Since a is relentlessly $\mathcal{B}$-reachable, there must be a state s' on π''' past s such that a is $\mathcal{B}$-reachable from s'. And since there are no occurrences of blocking actions on π''', a is also $\mathcal{B}$-reachable from s. Hence, a is $\mathcal{B}$-reachable from every state of π''' and is therefore perpetually $\mathcal{B}$-reachable on π'''.

We constructed π''' as a suffix of π'' which is a suffix of π', so π''' is a suffix of π' as well. We know that π' satisfies weak $\mathcal{B}$-hyperfairness, so since a is perpetually $\mathcal{B}$-reachable on π''', a suffix of π', a also occurs in π'''. Since π''' is a suffix of π'', we know that a occurs on π'. However, we assumed previously that a does not occur on π''. We have reached a contradiction and therefore conclude that π' satisfies strong $\mathcal{B}$-hyperfairness as well as weak $\mathcal{B}$-hyperfairness. ◀

▶ **Proposition 39.** *B-justness of actions is feasible.*

Proof. Let π be an arbitrary finite path. We prove π can be extended to a path π' satisfying $\mathcal{B}$-justness of actions. We do this through construction of such a path π'. We do this in steps, where π_i with $i \geq 0$ represents the path constructed in step i. Let $\pi_0 = \pi$. Let s_i be the last state of π_i for all $i \geq 0$. For this construction we use a queue Q.

The initial contents of Q are determined by π_0: it contains exactly one copy of every non-blocking action that is enabled in some state of π_0 but has not been subsequently eliminated. The order of these actions is arbitrary. The construction has the following invariant: Q contains exactly one copy of every non-blocking action that is enabled in some state of the path constructed so far, but has not been subsequently eliminated. Trivially, this invariant holds at initialisation.

The construction proceeds as follows: in step i, with $i > 0$, we construct π_i from π_{i-1} using Q. At this point, Q contains exactly one copy of every non-blocking action that was enabled in some state of π_{i-1} but has not subsequently been eliminated. If Q is empty, let $\pi_i = \pi_{i-1}$ and the construction terminates. Otherwise, we pop the head of Q, let this action be a. The action a must have been enabled in some state s_a of π_{i-1} such that the subpath π_a of π_{i-1} from s_a to s_{i-1} does not contain an occurrence of an action that eliminates a. By the second property of concurrency relations on actions, a must still be enabled in s_{i-1}. Let t_a be a transition enabled in s_{i-1} with $act(t_a) = a$, let $\pi_i = \pi_{i-1} t_a \, trgt(t_a)$. We modify Q in two steps: firstly, every action b that is in Q such that $b \not\smile^{\bullet} a$ is removed from Q. Secondly, every non-blocking action that is enabled in $trgt(t_a) = s_i$ that is not yet in Q gets appended to Q in some arbitrary order. At this point, the invariant is again satisfied: by removing all actions that are eliminated by a from Q, we ensure that Q no longer contains those actions that were not eliminated in π_{i-1} but are eliminated in π_i. By afterwards adding those actions that are enabled in s_i, we include those actions that are newly enabled without being eliminated yet. We proceed to the next iteration.

This construction either terminates after finitely many steps, or continues forever, the latter case results in an infinite path. We show that in either case, the constructed path π' satisfies $\mathcal{B}$-justness of actions.

- If the construction terminates during step i, then $\pi' = \pi_i$. By the invariant, Q contains exactly those non-blocking actions that are enabled in π' without being subsequently eliminated, and Q must be empty because the construction terminated. Let s' be the final state of π'. If there are non-blocking actions enabled in s', then those actions are enabled in π' without being subsequently eliminated, since there are no further transitions in π'. Since no such actions can exist, we know s' is a $\mathcal{B}$-locked state. If there were a non-blocking action enabled on π' that has not subsequently been eliminated through the occurrence of an interfering action, then by the second property of concurrency relations on actions that non-blocking action should still be enabled on s'. Since s' is a $\mathcal{B}$-locked state, this is impossible and hence π' is $\mathcal{B}$-JA.

- If the construction never terminates, then we construct an infinite path $\pi_\infty = \pi'$. Let s be some arbitrary state of π' and let a be an arbitrary non-blocking action that is enabled in s. We prove that a is eliminated in the suffix π'_a of π' starting in s. Towards a contradiction, assume that a is not eliminated in π'_a. Let $i > 0$ be the first iteration of the construction such that s_{i-1} is in π'_a. Since s_{i-1} is in π'_a, it either is s or comes after s, and since a is not eliminated in π'_a it must be the case, by the second property of concurrency relations on actions, that a is enabled in s_{i-1}. Hence, by the invariant, Q must have contained a at the start of iteration i. Since Q contains at most one copy of every non-blocking action and Act is finite, there are finitely many actions before a in the queue. Every

iteration, at least one action gets removed from Q and new actions get appended. Hence, a is either removed early or eventually becomes the head of the queue. If a is removed early, this is because an action occurred that eliminates a, hence a is eliminated in π'_a. If a becomes the head of the queue, then we ensure a itself occurs. By the first property of concurrency relations on actions, a eliminates itself. In this case too, a is eliminated in π'_a. This contradicts our assumption that a was not eliminated. We conclude that all non-blocking actions that are enabled in some state of π' are subsequently eliminated. Hence, π' satisfies $\mathcal{B}$-justness of actions.

In either case, the finite path π can be extended to a path π' that satisfies $\mathcal{B}$-JA. We conclude $\mathcal{B}$-justness of actions is feasible.   ◄

D   Correctness of Formulae

In this appendix, we provide the correctness proofs for the presented formulae. First, there is a supporting proposition we use repeatedly throughout the different proofs.

All our proofs assume a fixed LTSC $M = (\mathcal{S}, s_{init}, Act, Trans, \smallfrown\!\bullet)$, although the $\smallfrown\!\bullet$ is only relevant for JA. We also refer to an arbitrary environment e and set of blocking actions $\mathcal{B} \subseteq Act$. We define the length of a finite path to be the number of transitions occurring in it. A path of length 0 contains only a single state and is called the empty path.

D.1   Supporting Proposition

The following proposition gives the semantics of a least fixed point formula that occurs in several of our presented formulae.

▶ **Proposition 40.** *For all states $s \in \mathcal{S}$, formal variables Y, modal μ-calculus formulae ϕ_1 and ϕ_2 that do not depend on Y, and set of actions α, it is the case that s is in $[\![\mu Y.(\phi_1 \wedge (\phi_2 \vee \langle \alpha \rangle Y))]\!]$ if, and only if, s admits a finite path π satisfying the following requirements:*

1. *all actions occurring in π are in α, and*

2. *all states in π are in $[\![\phi_1]\!]$, and*

3. *the final state of π is in $[\![\phi_2]\!]$.*

We first need to prove a supporting lemma. To prove this lemma, we use an alternate characterisation of the semantics of least fixpoints to the one we presented in Section 2.2. We only give the definitions that we require for our proofs. The following presentation can be found in [8, 9], amongst others.

Let Y be an arbitrary formal variable, ϕ be an arbitrary modal μ-calculus formula and e an arbitrary environment. Let T be the transformer associated with $\mu Y.\phi$, defined as

$$T(\mathcal{F}) = \{s \in \mathcal{S} \mid s \in [\![\phi]\!]_{e[Y:=\mathcal{F}]}\}$$

And define

$$T^0(\mathcal{F}) = \mathcal{F}$$
$$T^{i+1}(\mathcal{F}) = T(T^i(\mathcal{F}))$$

Then we can calculate the semantics of $\mu Y.\phi$ under e as:

$$[\![\mu Y.\phi]\!]_e = \bigcup_{0 \leq i \leq |\mathcal{S}|} T^i(\emptyset)$$

Note that this definition only works for finite systems, since it uses $|\mathcal{S}|$. A version exists for infinite systems, but is not relevant here. We call $T^i(\emptyset)$ the i'th approximation of ϕ.

For the subsequent lemmas, we fix formal variable Y.

▶ **Lemma 41.** *For all environments e, states $s \in \mathcal{S}$, modal μ-calculus formulae ϕ_1 and ϕ_2 that do not depend on Y, sets of actions α, and natural numbers $0 \leq i \leq |\mathcal{S}|$, it holds that: s is in the i'th approximation of $\mu Y.(\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y))$ under e if, and only if, s admits a finite path π meeting the following conditions:*
1. *π has length at most $i - 1$, and*
2. *only actions in α occur in π, and*
3. *all states in π are in $[\![\phi_1]\!]_e$, and*
4. *the final state of π is in $[\![\phi_2]\!]_e$.*

Proof. Let T be the transformer of $\mu Y.(\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y))$. We prove that s is in $T^i(\emptyset)$ if, and only if, s admits a finite path π of length at most $i - 1$ on which only actions in α occur, all states are in $[\![\phi_1]\!]_e$, and which ends in a state $s' \in [\![\phi_2]\!]_e$. We do this by induction on i.

For the first *base*, take $i = 0$. Note that s is in the i'th approximation if $s \in T^0(\emptyset)$. However, $T^0(\emptyset) = \emptyset$, so s cannot be in the 0'th approximation. Indeed, we cannot have a path of length at most -1. So in both directions of the bi-implication, the left side of the implication does not hold.

For the second *base*, take $i = 1$. We prove the bi-implication.

- First, assume s is in the first approximation. Then $s \in T^1(\emptyset) = \{s \in \mathcal{S} \mid s \in [\![\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y)]\!]_{e[Y:=\emptyset]}\}$. Hence, $s \in [\![\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y)]\!]_{e[Y:=\emptyset]}$. Through the semantics of the modal μ-calculus, and using that ϕ_1 and ϕ_2 do not depend on Y, this becomes $s \in ([\![\phi_1]\!]_e \cap ([\![\phi_2]\!]_e \cup [\![\langle\alpha\rangle Y]\!]_{e[Y:=\emptyset]})$. Therefore, $s \in [\![\phi_1]\!]_e$, and s is in $[\![\phi_2]\!]_e$ or $s \in \{s \in \mathcal{S} \mid \exists_{s' \in \mathcal{S}}.s \xrightarrow{\alpha} s' \wedge s' \in \emptyset\}$. It is not possible for a state s' to exist that is in $\emptyset$, hence $s \in [\![\phi_2]\!]_e$. Let π be the empty path from s. No actions occur on π, so trivially all occurring actions are in α. Since s satisfies both ϕ_1 and ϕ_2 under e, the other conditions are met as well. We conclude that s admits a finite path of length at most 0 on which only actions in α occur, all states are in $[\![\phi_1]\!]_e$, and which ends in a state in $[\![\phi_2]\!]_e$.

- Second, assume s admits a path of length at most 0 on which only actions on α occur, all states are in $[\![\phi_1]\!]_e$, and which ends in a state in $[\![\phi_2]\!]_e$. The only path starting in s of length at most 0 is the path consisting of only s. Hence, $s \in [\![\phi_1]\!]_e$ and $s \in [\![\phi_2]\!]_e$. Since ϕ_1 and ϕ_2 do not depend on Y, we also have $s \in [\![\phi_1]\!]_{e[Y:=\emptyset]}$ and $s \in [\![\phi_2]\!]_{e[Y:=\emptyset]}$. If $s \in [\![\phi_2]\!]_{e[Y:=\emptyset]}$, it is also in the superset $[\![\phi_2]\!]_{e[Y:=\emptyset]} \cup [\![\langle\alpha\rangle Y]\!]_{e[Y:=\emptyset]}$. We conclude that $s \in \{s \in \mathcal{S} \mid s \in [\![\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y)]\!]_{e[Y:=\emptyset]}\} = T^1(\emptyset)$ and hence s is in the first approximation.

The *induction hypothesis* we use is that a state s' is in the k'th approximation of $\mu Y.(\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y))$ if, and only if, s' admits a finite path of length at most $k - 1$ on which only actions in α occur, all states are in $[\![\phi_1]\!]_e$, and which ends in a state in $[\![\phi_2]\!]_e$. This is for all $k \geq 1$.

For the *step* case, we prove the claim for $k + 1$. Let S be the set of states that admit finite paths of length at most $k - 1$ on which only actions in α occur, all states are in $[\![\phi_1]\!]_e$ and which end in a state in $[\![\phi_2]\!]_e$. By the induction hypothesis, $S = T^k(\emptyset)$. Since this lemma is a bi-implication, we prove both directions separately.

- We assume $s \in T^{k+1}(\emptyset)$. We need to prove s admits a path π that is of length at most $k + 1 - 1 = k$, on which only actions in α occur, all states are in $[\![\phi_1]\!]_e$, and which ends in a state in $[\![\phi_2]\!]_e$. We have $s \in T^{k+1}(\emptyset) = T(T^k(\emptyset)) = T(S)$ Hence, $s \in \{s \in \mathcal{S} \mid s \in [\![\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y)]\!]_{e[Y:=S]}\}$. This reduces to $s \in [\![\phi_1]\!]_e$, and $s \in [\![\phi_2]\!]_e \vee s \in$

$\{s \in \mathcal{S} \mid \exists_{s' \in \mathcal{S}}.s \xrightarrow{\alpha} s' \wedge s' \in S\}$, because ϕ_1 and ϕ_2 do not depend on Y. We do a case distinction on whether $s \in [\![\phi_2]\!]_e$.

- If $s \in [\![\phi_2]\!]_e$, then the path π consisting of only s is a path of length 0 on which only actions in α occur, and all states are in $[\![\phi_1]\!]_e$, and which ends in a state in $[\![\phi_2]\!]_e$. Since we assumed $k \geq 1$, we know $0 \leq k$, hence s admits a path meeting the requirements of length at most k.

- If $s \notin [\![\phi_2]\!]_e$, then $s \in \{s \in \mathcal{S} \mid \exists_{s' \in \mathcal{S}}.s \xrightarrow{\alpha} s' \wedge s' \in S\}$. Hence, there exists a state s' such that that there exists an α-transition t from s to s' and s' is in S. Since $s' \in S$, we know s' admits a path π' of length at most $k - 1$, on which only actions in α occur, all states are in $[\![\phi_1]\!]_e$, and which ends in a state satisfying $[\![\phi_2]\!]_e$. Let $\pi = st\pi'$. Since π' has length at most $k - 1$ and we added one transition, π has length at most k. Additionally, t is an α-transition, as are all transitions in π', so all transitions in π are labelled with actions in α. Since $s \in [\![\phi_1]\!]_e$ and all states in π' are as well, all states on π meet this requirement. Finally, since π' ends in a state satisfying $[\![\phi_2]\!]_e$, so does π. Hence, π is a witness that s admits a path meeting all requirements.

In both cases s admits such a path π of length at most k.

- We assume s admits a path π of length at most k such that all transitions on π are labelled with actions in α, all states are in $[\![\phi_1]\!]_e$, and π ends in a state satisfying $[\![\phi_2]\!]_e$. We prove $s \in T^{k+1}(\emptyset) = T(T^k(\emptyset)) = T(S) = \{s \in \mathcal{S} \mid s \in [\![\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y)]\!]_{e[Y:=S]}\}$. We do a case distinction on whether the length of π is zero.

  - If the length of π is zero, then $\pi = s$ and $s \in [\![\phi_2]\!]_e$. Additionally, since all states on π are in $[\![\phi_1]\!]_e$, so is s. Since ϕ_1 and ϕ_2 do not depend on Y, we also have $s \in [\![\phi_1]\!]_{e[Y:=S]}$ and $s \in [\![\phi_2]\!]_{e[Y:=S]}$, and hence also $s \in [\![\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y)]\!]_{e[Y:=S]}$. Thus, $s \in T(S) = T^{k+1}(\emptyset)$.

  - If the length of π is greater than zero, then there is at least one transition in π. Let t be the first transition of π. Since there are only α-transitions in π, t is an α-transition. Let s' be the target of t, and let π' be the suffix of π starting in s'. Then since the length of π is at most k, the length of π' is at most $k - 1$. Hence, π' witnesses that s' admits a path of length at most $k - 1$ on which only α-transitions occur, all states are in $[\![\phi_1]\!]_e$, and which ends in a state satisfying $[\![\phi_2]\!]_e$. Hence, $s \in S$. So s admits an α-transition, namely t, to a state in S, namely s'. Therefore $s \in [\![\langle\alpha\rangle Y]\!]_{e[Y:=S]}$ and hence also $s \in [\![\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y)]\!]_{e[Y:=S]}$. We conclude that $s \in T(S) = T^{k+1}(\emptyset)$.

  In both cases we demonstrate that s is in the $k + 1$'th approximation.

We have proven both sides of the bi-implication that s is in the $k + 1$'th approximation of $\mu Y.(\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y))$ if, and only if, s admits a finite path of length at most k on which only α actions occur, all states are in $[\![\phi_1]\!]_e$, and which ends in a state satisfying $[\![\phi_2]\!]_e$. This proves the step case.

By induction, we have proven the claim holds for all $i \geq 0$. Therefore it also holds for all $0 \leq i \leq |\mathcal{S}|$. We conclude the lemma holds.   ◄

We can now prove the main claim, Proposition 40:

▶ **Proposition 40.** *For all states $s \in \mathcal{S}$, formal variables Y, modal μ-calculus formulae ϕ_1 and ϕ_2 that do not depend on Y, and set of actions α, it is the case that s is in $[\![\mu Y.(\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y))]\!]$ if, and only if, s admits a finite path π satisfying the following requirements:*

1. *all actions occurring in π are in α, and*
2. *all states in π are in $[\![\phi_1]\!]$, and*
3. *the final state of π is in $[\![\phi_2]\!]$.*

Proof. This claim is a bi-implication, so we prove both directions.

- If s is in the semantics of $\mu Y.(\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y))$, then s is in the least fixed point of the transformer T matching this formula. Hence, there are one or more natural numbers $0 \leq i \leq |\mathcal{S}|$ such that $s \in T^i(\emptyset)$. Let i be the smallest such number, then by Lemma 41, s admits a path of length at most $i - 1$ that meets all three conditions. This path witnesses that s indeed admits a path meeting all three conditions.

- Assume s admits at least one finite path that satisfies all three conditions. Let π be the shortest such path that s admits. Let k be the length of π. We first prove that $0 \leq k + 1 \leq |\mathcal{S}|$. Trivially, a path has length at least 0, so $0 \leq k + 1$. Towards a contradiction, assume $k + 1 > |\mathcal{S}|$. A path of length j contains $j + 1$ individual instances of states: the initial state of the path and the target of every transition on the path. The path π has length k, and so contains at least $k + 1$ individual occurrences of states, and $k + 1 > |\mathcal{S}|$. Hence, π contains strictly more than $|\mathcal{S}|$ individual instances of states. Considering there are exactly $|\mathcal{S}|$ states in the LTS, by the pigeonhole principle there must be at least one state s' that is visited at least twice on π. Let π_1 be the prefix of π up until the first occurrence of s', and let π_2 be the suffix of π starting in the last occurrence of s'. Now let $\pi' = \pi_1 \cdot \pi_2$. This is a valid path, since π_1 ends in state s' and π_2 starts in this state. π' contains a subset of the actions and states of π, and has the same final state as π, so it satisfies all three conditions. Since we chose s' to be a state that occurred more than once on π, π' contains at least one transition less than π and hence $\pi \neq \pi'$. In fact, π' is shorter than π. However, we chose π to be the shortest path that meets all three conditions starting in s. We have reached a contradiction. Hence, we conclude that $k + 1 \leq |\mathcal{S}|$. This means that s admits a path of length at most k with $0 \leq k + 1 \leq |\mathcal{S}|$ that meets the three conditions. By Lemma 41, this means that s is in the $k + 1$'th approximation of $\mu Y.(\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y))$. Since the semantics of the formula are the union of all approximations, we conclude s is in $[\![\mu Y.(\phi_1 \wedge (\phi_2 \vee \langle\alpha\rangle Y))]\!]$.

We have proven both directions of the bi-implication. ◄

D.2 Proof of Progress Formula

We prove Theorem 21:

▶ **Theorem 21.** *A state in an LTS satisfies Formula 1 if, and only if, it does not admit $\mathcal{B}$-progressing paths that are $(\rho, \alpha_f, \alpha_e)$-violating.*

Formula 1 is:

$$\neg\langle\rho\rangle\nu X.(\langle\alpha_e\rangle tt \vee [\overline{\mathcal{B}}]ff \vee \langle\overline{\alpha_f}\rangle X)$$

We fix arbitrary $\mathcal{B}$, ρ, α_f and α_e for this proof. We first prove the formula without the $\neg\langle\rho\rangle$ at the start. For this, let S_P be the set of states that admit $\mathcal{B}$-progressing paths that are $(\varepsilon, \alpha_f, \alpha_e)$-violating. In other words, these are the states that admit paths that are $\mathcal{B}$-progressing and that are α_f-free up until an occurrence of an action in α_e.

We first prove that S_P is a fixed point of $\nu X.(\langle\alpha_e\rangle tt \vee [\overline{\mathcal{B}}]ff \vee \langle\overline{\alpha_f}\rangle X)$, and then that it is the greatest fixed point.

▶ **Lemma 42.** *S_P is a fixed point of the transformer T_P defined by:*

$$T_P(\mathcal{F}) = \bigcap_{a \in \overline{\mathcal{B}}} \{s \in \mathcal{S} \mid s \in [\![\langle\alpha_e\rangle tt \vee [\overline{\mathcal{B}}]ff \vee \langle\overline{\alpha_f}\rangle X]\!]_{e[X:=\mathcal{F}]}\}$$

Proof. To prove S_P is a fixed point of T_P, we prove $T_P(S_P) = S_P$. We do this through mutual set inclusion.

- Let s be an arbitrary state in $T_P(S_P)$. We therefore know that $s \in [\![\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}]ff \vee \langle \overline{\alpha_f} \rangle X]\!]_{e[X:=S_P]}$. We do a case distinction on which of those conditions s satisfies.

  - If s satisfies $\langle \alpha_e \rangle tt$, then there is a transition t enabled in s that is labelled with an action in α_e. Let $\pi = st\, trgt(t)$. This is a path that is α_f-free up until the first occurrence of an action in α_e. We now extend π arbitrarily, either until a $\mathcal{B}$-locked state is reached or infinitely. This is always possible: as long we are not in a $\mathcal{B}$-locked state there is always a non-blocking action enabled that can be appended to the path we are constructing. This way, a $\mathcal{B}$-progressing path that is $(\varepsilon, \alpha_f, \alpha_e)$-violating is constructed.

  - If s satisfies $[\overline{\mathcal{B}}]ff$, then s is a $\mathcal{B}$-locked state. Hence, the empty path is a path that s admits that is $\mathcal{B}$-progressing and on which, trivially, no actions in α_f occur. Hence, s admits a $\mathcal{B}$-progressing path that is $(\varepsilon, \alpha_f, \alpha_e)$-violating.

  - If s satisfies $\langle \overline{\alpha_f} \rangle X$ with $X = S_P$, then s admits a transition labelled with an action not in α_f to a state in S_P. Let t be such a transition and $s' = trgt(t)$. Then since $s' \in S_P$, s' admits a path π' that is $\mathcal{B}$-progressing and α_f-free up until the first occurrence of an action in α_e. Let $\pi = st\pi'$: this path too is $\mathcal{B}$-progressing and α_f-free up until the first occurrence of an action in α_e. Hence, π witnesses that s admits a $\mathcal{B}$-progressing path that is $(\varepsilon, \alpha_f, \alpha_e)$-violating.

  Therefore $s \in T_P(S_P) \Rightarrow s \in S_P$.

- Let s be an arbitrary state in S_P, then s admits a path π that is $\mathcal{B}$-progressing and α_f-free up until the first occurrence of α_e. We prove $s \in [\![\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}]ff \vee \langle \overline{\alpha_f} \rangle X]\!]_{e[X:=S_P]}$. First, we do a case distinction on whether π is the empty path.

  - If it is, then since π is $\mathcal{B}$-progressing, s must be a $\mathcal{B}$-locked state. Hence $s \in [\![[\overline{\mathcal{B}}]ff]\!]$.

  - If π is not the empty path, we do a case distinction on whether the first transition of π is labelled with an action in α_e.

    * If it is, then $s \in [\![\langle \alpha_e \rangle tt]\!]$.

    * If it is not, then since π is α_f-free up until the first occurrence of an action in α_e, and the first transition of π is not labelled with an action in α_e, this transition must be labelled with an action not in α_f. Let t be this transition and s' the target of t. Let π' be the suffix of π starting in s'. Then because π is $\mathcal{B}$-progressing and α_f-free up until the first occurrence of an action in α_e, π' meets those requirements as well. Hence, $s' \in S_P$. We conclude that s admits a non-α_f transition to a state in S_P, and so $s \in [\![\langle \overline{\alpha_f} \rangle X]\!]_{e[X:=S_P]}$.

  In every case, s is in one of the options of $[\![\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}]ff \vee \langle \overline{\alpha_f} \rangle X]\!]_{e[X:=S_P]}$. Hence, $s \in T_P(S_P)$.

By mutual set inclusion, we conclude that $T_P(S_P) = S_P$.   ◀

▶ **Lemma 43.** *S_P is the greatest fixed point of the transformer defined in Lemma 42.*

Proof. We prove S_P is the greatest fixed point by showing that for every $\mathcal{F} \subseteq \mathcal{S}$ such that $T_P(\mathcal{F}) = \mathcal{F}$, $\mathcal{F} \subseteq S_P$. To this end, let $\mathcal{F}$ be an arbitrary fixed point of T_P and let s be an arbitrary state in $\mathcal{F}$. We show that $s \in S_P$ by constructing a path π from s that is $\mathcal{B}$-progressing and α_f-free up until the first occurrence of an action in α_e.

We initialise the construction with $\pi = s$. Throughout the construction, let s' be the last state of the path π constructed thus far. The invariants of this construction are that $s' \in \mathcal{F}$, and the π constructed thus far is α_f-free. This trivially holds initially.

We know that $s' \in \mathcal{F}$ and $\mathcal{F} = T_P(\mathcal{F})$. Hence, $s' \in [\![\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}] ff \vee \langle \overline{\alpha_f} \rangle X]\!]_{e[X := \mathcal{F}]}$. We do a case distinction on which of the three cases s' satisfies.

- If $s' \in [\![\langle \alpha_e \rangle tt]\!]$ then s' admits a transition labelled with an action in α_e. Similar to the argument given in the proof of Lemma 42, we can use this transition as the first transition of an otherwise arbitrary path π' that is $\mathcal{B}$-progressing and, because its first transition is labelled with an action in α_e, α_f-free up until the first occurrence of an action in α_e. We append this path π' to the π constructed thus far and terminate the construction. The resulting path π is $\mathcal{B}$-progressing and α_f-free up until the first occurrence of an action in α_e, since before this iteration of the construction π was α_f-free. Hence, it witnesses $s \in S_P$.
- If $s' \in [\![[\overline{\mathcal{B}}] ff]\!]$ then s' is a $\mathcal{B}$-locked state and hence we can terminate with the π constructed thus far. This path is $\mathcal{B}$-progressing since it ends in a $\mathcal{B}$-locked state and by the invariant it is α_f-free. Hence, it witnesses $s \in S_P$.
- If $s' \in [\![\langle \overline{\alpha_f} \rangle X]\!]_{e[X := \mathcal{F}]}$ then s' admits a non-α_f-transition to a state in $\mathcal{F}$. We append this transition and its subsequent target to π and continue the construction. We know that the new last state of π must be in $\mathcal{F}$, and that the path is still α_f-free.

We have argued in the termination cases that the resulting path witnesses $s \in S_P$. If the construction never terminates, then we have an infinite path, which is therefore $\mathcal{B}$-progressing. By the invariant, the path will also be α_f-free, so it witnesses $s \in S_P$.

Since such a witness can always be constructed, $\mathcal{F} \subseteq S_P$ and so S_P is the greatest fixed point of T_P. ◀

▶ **Corollary 44.** *The set of states characterised by $\nu X.(\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}] ff \vee \langle \overline{\alpha_f} \rangle X)$ is exactly the set of states that admit $\mathcal{B}$-progressing paths that are $(\varepsilon, \alpha_f, \alpha_e)$-violating.*

All that remains is to prove that the ρ aspect of the $(\rho, \alpha_f, \alpha_e)$-violating path is properly characterised in the full formula.

▶ **Lemma 45.** *For all environments e and states $s \in \mathcal{S}$, it holds that $s \in [\![\langle \rho \rangle \nu X.(\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}] ff \vee \langle \overline{\alpha_f} \rangle X)]\!]_e$ if, and only if, s admits a path that is $\mathcal{B}$-progressing and $(\rho, \alpha_f, \alpha_e)$-violating.*

Proof. It follows directly from the definition of the diamond operator that this formula characterises those states that admit a path π that has a prefix matching ρ that ends in some state s' in $[\![\nu X.(\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}] ff \vee \langle \overline{\alpha_f} \rangle X)]\!]_e$. By Corollary 44, s' admits a $\mathcal{B}$-progressing path π' that is $(\varepsilon, \alpha_f, \alpha_e)$-violating. Hence, $\pi \cdot \pi'$ witnesses the admission of a $\mathcal{B}$-progressing and $(\rho, \alpha_f, \alpha_e)$-violating path. ◀

Formula 1 is the negation of this formula, hence it expresses that a state does not admit such a path. Theorem 21 follows directly.

D.3 Proof of WFA, WHFA and JA Formulae

In Section 6.2, we defined finitely realisable predicates as follows:

▶ **Definition 22.** *A path predicate P is* finitely realisable *if, and only if, there exist mappings ϕ_{on} and ϕ_{of} from non-blocking actions to closed modal μ-calculus formulae, and a mapping α_{el} from non-blocking actions to sets of actions, such that:*
1. *A path π satisfies predicate P if, and only if, all states s on π satisfy the following: for all $a \in \overline{\mathcal{B}}$, if s satisfies $\phi_{on}(a)$ then the suffix π' of π starting in s must contain an occurrence of some action in $\alpha_{el}(a)$ or a state that satisfies $\phi_{of}(a)$.*
2. *A state s is a $\mathcal{B}$-locked state if, and only if, $s \notin [\![\phi_{on}(a)]\!]$ for all $a \in \overline{\mathcal{B}}$.*

3. *For every state s and for all $a \in \overline{\mathcal{B}}$, $s \in \llbracket \phi_{on}(a) \rrbracket$ implies $s \notin \llbracket \phi_{of}(a) \rrbracket$.*
4. *For all states s and all $a \in \overline{\mathcal{B}}$ such that $s \in \llbracket \phi_{on}(a) \rrbracket$, if there exists a finite path π from s to a state s' such that there is no occurrence of an action in $\alpha_{el}(a)$ on π and there is no state on π that satisfies $\phi_{of}(a)$, then $s' \in \llbracket \phi_{on}(a) \rrbracket$.*

We refer to these four properties as the invariant property, *the* locking property, *the* exclusive property *and the* persistent property, *respectively.*

We gave a general formula for feasible, finitely-realisable completeness criteria, Formula 2:

$$\neg \langle \rho \rangle \nu X. (\bigwedge_{a \in \overline{\mathcal{B}}} (\phi_{on}(a) \Rightarrow \langle \overline{\alpha_f}^\star \rangle (\langle \alpha_e \rangle tt \vee (\phi_{of}(a) \wedge X) \vee \langle \alpha_{el}(a) \setminus \alpha_f \rangle X)))$$

We will prove the correctness of this formula before proving that it can indeed be used to express WFA, WHFA and JA. However, before proving anything on the general formula, we first prove a few properties that can be derived from those given in Definition 22.

▶ **Proposition 46.** *Every $\mathcal{B}$-progressing, finite path satisfies every finitely realisable path predicate, for all $\mathcal{B} \subseteq Act$.*

Proof. Let P be an arbitrary finitely realisable predicate on paths and let π be an arbitrary $\mathcal{B}$-progressing, finite path. We assume towards a contradiction that π does not satisfy P. Then there must exist a state s on π and an action $a \in \overline{\mathcal{B}}$ such that s satisfies $\phi_{on}(a)$, and the suffix of π starting in s, π', does not contain an occurrence of an action in $\alpha_{el}(a)$ nor a state that satisfies $\phi_{of}(a)$. Since π is finite, so is π'. Let s_f be the final state of π'. By the persistent property, s_f satisfies $\phi_{on}(a)$. However, by the locking property that means s_f cannot be a $\mathcal{B}$-locked state. Since s_f is also the final state of π, this means π is not a $\mathcal{B}$-progressing path. This contradicts the assumption on π; we conclude that π satisfies P. ◀

▶ **Proposition 47.** *Every path that satisfies a finitely realisable path predicate P is $\mathcal{B}$-progressing.*

Proof. Let P be an arbitrary finitely realisable predicate on paths and let π be an arbitrary path. If π is infinite, then it is trivially $\mathcal{B}$-progressing. Hence, we assume π is finite. Let s be the final state of π. Towards a contradiction, assume s is not $\mathcal{B}$-locked. Then by the locking property, there must exist an action $a \in \overline{\mathcal{B}}$ such that $s \in \llbracket \phi_{on}(a) \rrbracket$. Since π satisfies P, it must be the case that in the suffix π' of π starting in s, an action in $\alpha_{el}(a)$ occurs or a state occurs that satisfies $\phi_{of}(a)$, by the invariant property. The suffix of π starting in its final state s consists of only the state s itself, so trivially there are no action occurrences. So it must be the case that $s \in \llbracket \phi_{of}(a) \rrbracket$. However, by the exclusive property this is impossible. We have reached a contradiction and conclude that s is a $\mathcal{B}$-locked state, and hence π is $\mathcal{B}$-progressing. ◀

▶ **Proposition 48.** *If a path π satisfies finitely realisable path predicate P, then every path of which π is a suffix also satisfies P.*

Proof. Let P be an arbitrary finitely realisable predicate on paths and let π be an arbitrary path that satisfies P, with initial state s_0. Let π' be an arbitrary finite path that ends in s_0. We prove that $\pi'' = \pi' \cdot \pi$ satisfies P. Towards a contradiction, assume π'' does not satisfy P. Then there must exist a state s on π'' such that s satisfies $\phi_{on}(a)$ for some $a \in \overline{\mathcal{B}}$, yet the suffix of π'' starting in s, π_s, contains no occurrence of an action in $\alpha_{el}(a)$ nor a state that satisfies $\phi_{of}(a)$. If s is on π, then since π satisfies P this situation is impossible. Hence,

we must assume s is on π'. Let π'_s be the subpath of π'' from s to s_0, this is a prefix of π_s. Since π_s contains no occurrence of an action in $\alpha_{el}(a)$ or a state in $\phi_{of}(a)$, the same holds for π'_s. By the persistent property, that means s_0 satisfies $\phi_{on}(a)$. Since s_0 is on π, and π satisfies P, this means that π must contain an occurrence of an action in $\alpha_{el}(a)$ or a state in which $\phi_{of}(a)$ is satisfied. Since π is a suffix of π_s as well, π_s contains such an occurrence too. This contradicts that π_s does not contain such an occurrence. Hence, we conclude π'' satisfies P. ◀

▶ **Proposition 49.** *If a path π satisfies a finitely realisable path predicate P, then every suffix of π also satisfies P.*

Proof. This follows directly from the invariant property: a path π satisfies an finitely realisable predicate on paths P exactly when every state on π meets the condition that all non-blocking actions for which the state satisfies $\phi_{on}(a)$ are eliminated (through $\alpha_{el}(a)$ or $\phi_{of}(a)$) in the suffix of the path starting in that state. If this condition holds on all states of a path, it also holds on all states of a suffix of that path. ◀

We now prove Theorem 24:

▶ **Theorem 24.** *For all feasible and finitely realisable path predicates P, it holds that an LTSC satisfies Formula 2 if, and only if, its initial state does not admit $\mathcal{B}$-progressing paths that satisfy P and are $(\rho, \alpha_f, \alpha_e)$-violating.*

For the proof of this theorem, we fix $\mathcal{B}$, ρ, α_f and α_e. We also fix a feasible, finitely realisable path predicate P. To characterise the semantics of the formula, we first split it into multiple smaller subformulae.

$$
\begin{aligned}
violate_G &= \langle \rho \rangle invariant_G \\
invariant_G &= \nu X.(\bigwedge_{a \in \overline{\mathcal{B}}} (\phi_{on}(a) \Rightarrow eliminate_G(a))) \\
eliminate_G(a) &= \langle \overline{\alpha_f}^\star \rangle (\langle \alpha_e \rangle tt \vee (\phi_{of}(a) \wedge X) \vee \langle \alpha_{el}(a) \setminus \alpha_f \rangle X)
\end{aligned}
$$

We have that Formula $2 = \neg violate_G$.

We can now characterise the semantics of parts of the formula. We start with $eliminate_G$.

▶ **Lemma 50.** *For all environments e, states $s \in \mathcal{S}$, actions $a \in \overline{\mathcal{B}}$ and sets $\mathcal{F} \subseteq \mathcal{S}$, it holds that $s \in [\![eliminate_G(a)]\!]_{e[X:=\mathcal{F}]}$ if, and only if, s admits a finite path π with final state s_{final} that satisfies the following conditions:*
1. *π is α_f-free, and*
2. *one of the following three holds:*
    a. *at least one action in α_e is enabled in s_{final}, or*
    b. *$s_{final} \in \mathcal{F}$ and s_{final} satisfies $\phi_{of}(a)$, or*
    c. *$s_{final} \in \mathcal{F}$ and the last transition in π, t_{final}, is labelled with an action in $\alpha_{el}(a) \setminus \alpha_f$.*

Proof. Note that the formula $eliminate_G(a)$ is equivalent to $\mu Y.(\langle \alpha_e \rangle tt \vee (\phi_{of}(a) \wedge X) \vee \langle \alpha_{el}(a) \setminus \alpha_f \rangle X \vee \langle \overline{\alpha_f} \rangle Y)$, since the regular formula hides the least fixpoint operator. Since $\phi_{of}(a)$ is a closed formula, $\langle \alpha_e \rangle tt \vee (\phi_{of}(a) \wedge X) \vee \langle \alpha_{el}(a) \setminus \alpha_f \rangle X$ is a formula that does not depend on Y. We can therefore apply Proposition 40 (with $\phi_1 = tt$) to conclude that $s \in [\![eliminate_G(a)]\!]_{e[X:=\mathcal{F}]}$ if, and only if, s admits a finite path π' meeting the following conditions:

3. all actions occurring in π' are in $\overline{\alpha_f}$, and

4. the final state of π', s'_{final} is in $[\![\langle \alpha_e \rangle tt \vee (\phi_{of}(a) \wedge X) \vee \langle \alpha_{el}(a) \setminus \alpha_f \rangle X]\!]_{e[X:=\mathcal{F}]}$.

Condition 3 can be restated as π' being α_f-free. Condition 4 can be subdivided into three options: at least one action in α_e is enabled in s'_{final} (4a), or $s'_{final} \in \mathcal{F}$ and s'_{final} satisfies $\phi_{of}(a)$ (4b), or s'_{final} admits a transition to a state in $\mathcal{F}$ that is labelled with an action in $\alpha_{el}(a) \setminus \alpha_f$ (4c). The correspondence between 1 and 3, as well as 2a and 4a, and 2b and 4b, is direct. In the a and b cases, in both directions, π and π' describe paths meeting the exact same conditions. It remains to argue that if s admits a path π meeting conditions 1 and 2c, it also admits a path π' meeting conditions 3 and 4c, and vice versa. Here too, the correspondence is rather direct: π contains at least one transition, its last transition, which is labelled with an action in $\alpha_{el}(a) \setminus \alpha_f$ to a state in $\mathcal{F}$. A path π' can be constructed from π by dropping this last transition. The other way around, 4c witnesses such a transition exists, so by appending this transition to π' a path π can be constructed. Hence, it follows from Proposition 40 that s indeed admits a path meeting conditions 1 and 2 exactly when it is in the semantics of $eliminate_G(a)$ when $X = \mathcal{F}$.   ◀

We proceed to characterise the semantics of $invariant_G$. We will prove that it exactly describes those states that admit paths that are $\mathcal{B}$-progressing, $(\varepsilon, \alpha_f, \alpha_e)$-violating and satisfy P. Formally, define set S_G to be exactly those states in M that admit a path π meeting the following conditions:

- π satisfies P, and
- π is $\mathcal{B}$-progressing, and
- π satisfies one of the following conditions:
  - π is α_f-free, or
  - π contains an occurrence of an action in α_e, and the prefix of π before the first occurrence of an action in α_e is α_f-free.

As before, we frequently refer to the last condition as π being α_f-free up until the first occurrence of an action in α_e.

We prove that $[\![invariant_G]\!]_e = S_G$. We do this by proving S_G is the greatest fixed point of the transformer characterising $invariant_G$. To do this, we first prove it is a fixed point of this transformer and then that every fixed point of this transformer is a subset of S_G.

▶ **Lemma 51.** S_G *is a fixed point of the transformer* T_G *defined by:*

$$T_G(\mathcal{F}) = \bigcap_{a \in \overline{\mathcal{B}}} \left\{ s \in \mathcal{S} \mid s \in [\![\phi_{on}(a)]\!]_{e[X:=\mathcal{F}]} \Rightarrow s \in [\![eliminate_G(a)]\!]_{e[X:=\mathcal{F}]} \right\}$$

Proof. To prove S_G is a fixed point of T_G, we need to prove $T_G(S_G) = S_G$. We do this through mutual set inclusion.

- Let s be an arbitrary element of $T_G(S_G)$, we prove $s \in S_G$. From $s \in T_G(S_G)$ we know that for all actions $a \in \overline{\mathcal{B}}$, $s \in [\![\phi_{on}(a)]\!]_{e[X:=S_G]}$ implies $s \in [\![eliminate_G(a)]\!]_{e[X:=S_G]}$. We do a case distinction on whether s satisfies $\phi_{on}(a)$ for some $a \in \overline{\mathcal{B}}$.
  - If there are no non-blocking actions a such that s satisfies $\phi_{on}(a)$, then by the locking property it must be the case that s is a $\mathcal{B}$-locked state. Hence, trivially s admits a $\mathcal{B}$-progressing path on which no actions in α_f ever occur, namely the empty path. By Proposition 46, it must be the case that a finite, $\mathcal{B}$-progressing path satisfies P. Hence, $s \in S_G$.
  - If there is such an action, let a be an arbitrary non-blocking action such that s satisfies $\phi_{on}(a)$. Because $s \in [\![\phi_{on}(a)]\!]_{e[X:=S_G]}$, $s \in [\![eliminate_G(a)]\!]_{e[X:=S_G]}$. By Lemma 50,

we conclude that s admits a finite path π with final state s_{final} meeting the following requirements:

1. π is α_f-free, and
2. one of the following three holds:
   a. at least one action in α_e is enabled in s_{final}, or
   b. $s_{final} \in S_G$ and s_{final} satisfies $\phi_{of}(a)$, or
   c. $s_{final} \in S_G$ and the last transition in π, t_{final} is labelled with an action in $\alpha_{el}(a) \setminus \alpha_f$.

We use this information to construct a path π_w that witnesses $s \in S_G$. For this, we do a case distinction on whether 2a holds.

* If π satisfies 1 and 2a, then π is a path on which no actions in α_f occur until a state is reached where at least one action in α_e is enabled. Let π' be π extended with an arbitrary transition labelled with an action in α_e and the subsequent target state. Then π' is a finite path on which there are no occurrences of actions in α_f before the first occurrence of an action in α_e. This path is either already $\mathcal{B}$-progressing or not. In the former case, it is finite and therefore trivially satisfies P (Proposition 46). We can then take $\pi' = \pi_w$. Alternatively, it is not $\mathcal{B}$-progressing. In this case, then because P is feasible we can extend it to a path π_w that satisfies P, and therefore by Proposition 47 is also $\mathcal{B}$-progressing. On π_w, it is still the case that there are no occurrences of actions in α_f until the first occurrence of an action in α_e. Hence, we have provided a witness for $s \in S_G$.

* If π does not satisfy 2a, then it satisfies 1 and 2b, or 1 and 2c. In either case, s admits a finite, α_f-free path to a state s_{final}. Since s_{final} is in S_G, we know it admits a $\mathcal{B}$-progressing path π' that satisfies P and is α_f-free up until the first occurrence of an action in α_e. Let $\pi_w = \pi \cdot \pi'$. Since π is α_f-free, we know that prepending π to π' does not violate that the resulting path is α_f-free until the first occurrence of an action in α_e. Additionally, by Proposition 48, π_w satisfies P because π' does. Similarly, π_w is $\mathcal{B}$-progressing because π' is. Hence, π_w witnesses $s \in S_G$.

We have proven in every scenario that $s \in S_G$.

- Assume $s \in S_G$, we prove $s \in T_G(S_G)$. To prove $s \in T_G(S_G)$, we need to show that for all non-blocking actions a such that s satisfies $\phi_{on}(a)$, $s \in [\![eliminate_G(a)]\!]_{e[X:=S_G]}$. Towards this end, we prove $s \in [\![eliminate_G(a)]\!]_{e[X:=S_G]}$, where a is an arbitrary non-blocking action such that $\phi_{on}(a)$ is satisfied in s. By Lemma 50, it is sufficient to prove that s admits a finite path π with final state s_{final} satisfying the the same two conditions as given previously as (1) and (2). We know from $s \in S_G$ that s admits a $\mathcal{B}$-progressing path π_G that satisfies P and is α_f-free up until the first occurrence of an action in α_e. We use π_G to construct π.

First, we do a case distinction on whether there is an occurrence of an action in α_e in π_G.

- If there is, then let t_{en} be the first transition of π_G that is labelled with an action in α_e. Let s_{final} be the source state of t_{en}. Then we take π to be the prefix of π_G ending in s_{final}. Since π_G is α_f-free up until the first occurrence of an action in α_e, and t_{en} is the first occurrence of action action in α_e, we know π is α_f-free, so satisfies condition 1. Additionally, by construction an action in α_e is enabled in s_{final}, the final state of π, because t_{en} is enabled. Therefore, π satisfies condition 2a. We have constructed a path π meeting conditions 1 and 2.

- If there is no occurrence of an action in α_e in π_G, then π_G is fully α_f-free. Recall that we assumed that s satisfies $\phi_{on}(a)$ for the non-blocking action a. Since π_G satisfies P, we know by the invariant property that π_G contains an occurrence of some action in

$\alpha_{el}(a)$ or a state that satisfies $\phi_{of}(a)$. We do a case distinction on whether some state of π_G satisfies $\phi_{of}(a)$.

∗ If π_G contains a state satisfying $\phi_{of}(a)$, let s_{final} be the first such state and let π be the prefix of π_G ending in s_{final}. Then π is α_f-free because π_G is, and ends in a state satisfying $\phi_{of}(a)$. To show π satisfies conditions 1 and 2b, it remains to show that s_{final} is in S_G. By Proposition 49, the suffix π'_G of π_G starting in s_{final} satisfies P. Additionally, π'_G is $\mathcal{B}$-progressing and α_f-free because π_G is. Hence, π'_G witnesses $s_{final} \in S_G$. The π we have constructed satisfies conditions 1 and 2.

∗ If no state of π_G satisfies $\phi_{of}(a)$, then π_G must contain an occurrence of some action in $\alpha_{el}(a)$. Then we let t be the first transition that occurs in π_G that is labelled with an action in $\alpha_{el}(a)$. Let $act(t) = b$. Note that because π_G is α_f-free, $b \in \alpha_{el}(a) \setminus \alpha_f$. Let π be the prefix of π_G such that t is the final transition of π. Then π is α_f-free (condition 1) and the last transition of π is labelled with an action in $\alpha_{el}(a) \setminus \alpha_f$. To show this π satisfies both 1 and 2c, it remains to show that the final state of π, s_{final} is in S_G. The same argument applies here as in the previous case. Hence, π satisfies conditions 1 and 2.

We have proven $s \in [\![eliminate_G(a)]\!]_{e[X:=S_G]}$ for all a such that s satisfies $\phi_{on}(a)$, and hence $s \in T_G(S_G)$.

By mutual set inclusion, S_G is a fixed point of T_G. ◄

Next, we need to prove S_G is the greatest fixed point of the transformer. For this, we first need a supporting lemma.

▶ **Lemma 52.** *For all states s in a fixed point $\mathcal{F}$ of T_G as defined in Lemma 51, if there is no action in α_e that is reachable from s without doing an action in α_f and there exists at least one action $a \in \overline{\mathcal{B}}$ such that s satisfies $\phi_{on}(a)$, then there exists a finite path π from s to some state s' meeting all of the following conditions:*

1. *$s' \in \mathcal{F}$, and*
2. *π has length at least one, and*
3. *π is α_f-free, and*
4. *for all actions $a \in \overline{\mathcal{B}}$ such that s satisfies $\phi_{on}(a)$, there is a state on π that satisfies $\phi_{of}(a)$ or there is a transition on π labelled with an action in $\alpha_{el}(a)$.*

Proof. Let $\mathcal{F}$ be an arbitrary fixed point of T_G and let s be an arbitrary state in $\mathcal{F}$ such that there is at least one non-blocking action for which s satisfies ϕ_{on} and it is impossible to reach a state in which an action in α_e is enabled from s without doing an action in α_f. Let L be the set of non-blocking actions for which s satisfies ϕ_{on}. By the definition of s, we have $|L| \geq 1$.

We prove the more general claim that for all subsets $L' \subseteq L$ that contain at least one element, s admits a path π meeting conditions 1 through 3 as well as the following rephrasing of condition 4 with respect to L': for all actions $a \in L'$, there is a state on π that satisfies $\phi_{of}(a)$ or there is a transition on π labelled with an action in $\alpha_{el}(a)$. We prove this by induction on the size of L'. Since L contains at least one element and is a subset of itself, this also proves the lemma.

Base case: for $|L'| = 1$, let a be the one non-blocking action in L'. Since $L' \subseteq L$, s satisfies $\phi_{on}(a)$. Observe that since $s \in \mathcal{F}$ and $\mathcal{F} = T_G(\mathcal{F})$, $s \in [\![\phi_{on}(a)]\!]_{e[X:=\mathcal{F}]} \Rightarrow s \in [\![eliminate_G(a)]\!]_{e[X:=\mathcal{F}]}$. By Lemma 50, this allows us to conclude that s admits a finite path π to some state s' such that the following holds:

5. *π is α_f-free, and*

6. one of the following three holds:

   a. at least one action in α_e is enabled in s', or

   b. $s' \in \mathcal{F}$ and s' satisfies $\phi_{of}(a)$, or

   c. $s' \in \mathcal{F}$ and the last transition of π, t, is labelled with an action in $\alpha_{el}(a) \setminus \alpha_f$.

We prove that this π also satisfies conditions 1 through 4. We assumed that it is impossible to reach a state in which an action in α_e is enabled from s without doing an action in α_f. Since π is α_f-free, it is therefore impossible that an action in α_e is enabled in s'. We can conclude that condition 6a cannot hold. Hence, 6b or 6c must hold. In either case, $s' \in \mathcal{F}$ so condition 1 is satisfied. If condition 6b holds, then s' satisfies $\phi_{of}(a)$. We know that s satisfies $\phi_{on}(a)$ and by the exclusive property, this means s and s' cannot be the same state so condition 2 holds. If 6c holds instead, then since π contains at least one transition it has length at least one, so condition 2 also holds. Condition 3 follows directly from condition 5. It remains to show that condition 4 is satisfied, regardless of whether 6b or 6c holds. This follows from a being the only action in L' and, either through 6b or 6c, there being some occurrence of a state satisfying $\phi_{of}(a)$ or an occurrence of an action in $\alpha_{el}(a)$. We conclude that this π meets conditions 1 through 4.

Step case: let $|L'| = n + 1$ for $n \geq 1$ and assume that the claim holds when $|L'| = n$ (*induction hypothesis*). We prove that s has a path π to a state s' meeting all four conditions. Let a be an arbitrary action in L'. Now apply the induction hypothesis to s with respect to the set $L'' = L' \setminus \{a\}$, which has size n. This gives us a path π' to a state s'' meeting all four conditions, condition 4 specifically with respect to L''. We use π' to construct π. We do a case distinction on whether s'' satisfies $\phi_{on}(a)$.

▪ If it does, then since $s'' \in \mathcal{F}$ and $\mathcal{F}$ is a fixed point of T_G, $s'' \in [\![eliminate_G(a)]\!]_{e[X:=\mathcal{F}]}$. We can then use the same argument as given in the base case and apply Lemma 50 to find a path π'' from s'' to some state $s' \in \mathcal{F}$ that is α_f-free and on which an action in $\alpha_{el}(a)$ occurs or on which a state exists that satisfies $\phi_{of}(a)$. Let $\pi = \pi' \cdot \pi''$; this π satisfies all four conditions. That $s' \in \mathcal{F}$ follows from the application of Lemma 50; we know from the application of the induction hypothesis that π' has length at least one, so π does as well; both π' and π'' are α_f-free; and finally for all actions in L' except a there is a state in which ϕ_{of} is satisfied for this action, or an action in the associated α_{el} occurs in π', the action a itself is eliminated in π''. Hence, we eliminate all actions in L'.

▪ If s'' does not satisfy $\phi_{of}(a)$, then by the persistent property, it must be the case that π' already contains a state that satisfies $\phi_{of}(a)$ or an occurrence of an action in $\alpha_{el}(a)$. Hence, π' already satisfies all four conditions for the whole set L' and so is a candidate for π.

In either case, there exists a path π meeting all four conditions. This proves the claim on arbitrary subsets of L containing at least one element. As stated, L is a subset of itself and contains at least one element, and so the lemma follows directly. ◀

▶ **Lemma 53.** *S_G is the greatest fixed point of the transformer T_G as defined in Lemma 51.*

Proof. Let $\mathcal{F} \subseteq \mathcal{S}$ be an arbitrary fixed point of T_G, meaning we have $\mathcal{F} = T_G(\mathcal{F})$. We prove that $\mathcal{F} \subseteq S_G$. To this end, let s be an arbitrary state in $\mathcal{F}$. We prove $s \in S_G$ by constructing a path π from s that is $\mathcal{B}$-progressing, satisfies P and on which no actions in α_f occur up until the first occurrence of an action in α_e.

We do a case distinction on whether there exists some non-blocking action a such hat s satisfies $\phi_{on}(a)$. If no such action exists, then by the locking property, s is a $\mathcal{B}$-locked state. In this case, the empty path is a $\mathcal{B}$-progressing from s which, by Proposition 46, satisfies P and on which trivially no actions in α_f occur. This path witnesses $s \in S_G$.

We proceed under the assumption that there exists some non-blocking action a such that s satisfies $\phi_{on}(a)$. Let a be an arbitrary such action. Since $s \in \mathcal{F}$ and $\mathcal{F}$ is a fixed point of T_G, we conclude that $s \in [\![eliminate_G(a)]\!]_{e[X:=\mathcal{F}]}$. By Lemma 50 we know that s admits a finite path π' to a state s_{final} such that the following holds:

1. π' is α_f-free, and
2. one of the following holds:
    - at least one action in α_e is enabled in s_{final}, or
    - $s_{final} \in \mathcal{F}$ and s_{final} satisfies $\phi_{of}(a)$, or
    - $s_{final} \in \mathcal{F}$ and the last transition in π', t_{final} is labelled with an action in $\alpha_{el}(a) \setminus \alpha_f$.

We do a case distinction on whether s admits some path π' that meets conditions 1 and 2a. If it does, then this π' is a finite, α_f-free path to a state in which an action in α_e is enabled. We extend π' by appending a transition labelled with an action in α_e to it together with the associated target state, creating a finite path π'' that is α_f-free up until the first occurrence of an action in α_e. By feasibility of P, we can extend π'' to a path π that satisfies P and, by Proposition 47 is also $\mathcal{B}$-progressing. It still holds that this path is α_f-free up until the first occurrence of an action in α_e. Hence, this π witnesses $s \in S_G$.

If s does not admit a path that satisfies conditions 1 and 2a, then it is impossible to reach a state in which an action in α_e is enabled from s without doing α_f-steps. We can then apply Lemma 52 to conclude that s admits a finite path π' to a state s' that meets the following requirements:

1. $s' \in \mathcal{F}$, and
2. π' has length at least one, and
3. π' is α_f-free, and
4. for all action $a \in \overline{\mathcal{B}}$ such that s satisfies $\phi_{on}(a)$, there is a state on π' that satisfies $\phi_{of}(a)$ or there is a transition on π' labelled with an action in $\alpha_{el}(a)$.

We can use this to construct π from s. We start with $\pi = \pi'$. Subsequently, we consider the last state of the path constructed thus far, s': if there are no actions $a \in \overline{\mathcal{B}}$ such that s' satisfies $\phi_{on}(a)$, then by the locking property the constructed path is $\mathcal{B}$-progressing and finite, and hence by Proposition 46 it satisfies P. Additionally, since it is constructed from α_f-free segments it is α_f-free and hence witnesses $s \in S_G$. If there are non-blocking actions for which s' satisfies ϕ_{on}, then we can apply Lemma 52 again to find a new π' and append it to the path constructed thus far. The procedure then repeats, potentially infinitely. Note that Lemma 52 can always be applied because the last state of the constructed path is always in $\mathcal{F}$, and we have established already that it is impossible to reach a state in which an action in α_e is enabled from s without doing an action in α_f: this remains true for all states reached through appending α_f-free segments to π.

If the construction ever terminates then it is because we reached a state where no actions are "on". Hence, as argued above, we have constructed a path that witnesses $s \in S_G$. If the procedure repeats infinitely then an infinite path is constructed, this is because every segment we add has length at least one. If an infinite path π is constructed this way, then this path is trivially $\mathcal{B}$-progressing. It is also α_f-free, since it is constructed from α_f-free segments. It remains to argue π satisfies P. To this end, let s_* be an arbitrary state of π and let a be an arbitrary non-blocking action such that s_* satisfies $\phi_{on}(a)$. Let π_* be the suffix of π starting in s_*. Towards a contradiction, assume that π_* contains neither a state in which $\phi_{of}(a)$ is satisfied, nor a transition labelled with an action in $\alpha_{el}(a)$. Then by the persistent property, every state of π_* must satisfy $\phi_{on}(a)$. In our construction, we always append finite paths to the infinite path we are constructing. Hence, there exist infinitely many states of π_* to which Lemma 52 has been applied to find the next path to append to

π. Let s'_* be such a state. Since $\phi_{on}(a)$ must be satisfied in s'_*, it must be the case that the path we added to π after applying Lemma 52 to s'_* contained a state that satisfies $\phi_{of}(a)$ or a transition labelled with an action in $\alpha_{el}(a)$. This contradicts our assumption that neither such a state nor such a transition exists in π_*. Hence, we conclude that the infinite path π satisfies P and thus witnesses $s \in S_G$.

In every case, $s \in S_G$ and thus $\mathcal{F} \subseteq S_G$, therefore S_G is the greatest fixed point of T_G. ◄

Since the semantics of $invariant_G$ are exactly the greatest fixed point of T_G, we can conclude the following from the definition of S_G.

▶ **Corollary 54.** *The set of states characterised by $invariant_G$ is exactly the set of states that admit $\mathcal{B}$-progressing, $(\varepsilon, \alpha_f, \alpha_e)$-violating paths that satisfy P.*

All that remains is to prepend the ρ part of the formula.

▶ **Lemma 55.** *For all environments e and states $s \in \mathcal{S}$, it holds that $s \in [\![violate_G]\!]_e$ if, and only if, s admits a path that is $\mathcal{B}$-progressing, satisfies P and is $(\rho, \alpha_f, \alpha_e)$-violating.*

Proof. It follows directly from the definition of the diamond operator that $violate_G$ characterises those states that admit a path π that has a prefix matching ρ that ends in some state s' in $[\![invariant_G]\!]_e$. By Corollary 54, s' admits a $\mathcal{B}$-progressing path π' that is α_f-free up until the first occurrence of an action in α_e and satisfies P. By Proposition 48, prepending π to π' results in a path that still satisfies P. It is also still $\mathcal{B}$-progressing, and by construction, it has a prefix matching ρ, namely π, after which it is α_f-free up until the first occurrence of an action in α_e. Hence, it is $(\rho, \alpha_f, \alpha_e)$-violating. ◄

Formula 2 is the negation of $violate_G$, hence it expresses that a state does not admit such a path. Theorem 24 follows directly.

In Section 6.2, we described how ϕ_{on}, ϕ_{of} and α_{el} should be defined for WFA, WHFA and JA. We here prove that these assignments indeed satisfy the conditions on finitely realisable predicates, and then use Theorem 24 to prove the formulae correct.

D.3.1 WFA

We prove Corollary 25.

▶ **Corollary 25.** *A state in an LTS satisfies Formula 2 with $\phi_{on}(a) = \langle a \rangle tt$, $\phi_{of}(a) = [a]ff$ and $\alpha_{el}(a) = \{a\}$ for all $a \in \overline{\mathcal{B}}$ if, and only if, it does not admit $\mathcal{B}$-progressing paths that satisfy $\mathcal{B}$-weak fairness of actions and are $(\rho, \alpha_f, \alpha_e)$-violating.*

Proof. We argue this assignment to ϕ_{on}, ϕ_{of} and α_{el} satisfies the four properties of finitely realisable predicates.

1. The invariant property: let π be an arbitrary path. According to $\mathcal{B}$-WFA, π is complete if for all suffixes of π, all non-blocking actions that are perpetually enabled on the suffix also occur in that suffix. We prove this is equivalent to the condition that a path π is complete if for all states s in π and all non-blocking actions a, if s satisfies $\langle a \rangle tt$ then the suffix of π starting in s contains a state satisfying $[a]ff$ or an occurrence of a. Refer to this condition as P. Let π' be an arbitrary suffix of π and let s be the first state of π'. Let a be an arbitrary non-blocking action. If a is not enabled in s, then WFA does not require a to occur in π' since it is not perpetually enabled, and P does not require a to occur in π' because $\langle a \rangle tt$ is not satisfied in s. If a is enabled in s, we consider if a is

perpetually enabled on π'. If it is not, WFA does not require a to occur in π'. Also P does not require this, since within finitely many steps a state will occur on π' in which $[a]f\!f$ is satisfied. If a is perpetually enabled in π', then WFA requires a to occur on π'. Since there are no states on π' that satisfy $[a]f\!f$, P also requires an occurrence of a in π'. Hence, in all cases, WFA and P give the exact same requirements on action occurrences in π'.

2. The locking property: if a state is $\mathcal{B}$-locked, no non-blocking actions are enabled in that state. This is identical to the definition that $\langle a \rangle tt$ is violated for all $a \in \overline{\mathcal{B}}$.

3. The exclusive property: $\langle a \rangle tt = \neg[a]f\!f$ for all $a \in Act$, so this follows directly.

4. The persistent property: this follows from $\langle a \rangle tt = \neg[a]f\!f$ as well. If there is no state on a path that satisfies $[a]f\!f$ for some $a \in \overline{\mathcal{B}}$, then the final state of that path satisfies $\neg[a]f\!f$ and hence satisfies $\langle a \rangle tt$.

Hence, $\mathcal{B}$-WFA is a finitely realisable predicate on paths.

That $\mathcal{B}$-WFA is feasible was proven as Proposition 34. We apply Theorem 24 to conclude the theorem holds. ◄

D.3.2 WHFA

We prove Corollary 26.

▶ **Corollary 26.** *A state in an LTS satisfies Formula 2 with $\phi_{on}(a) = \langle \overline{\mathcal{B}}^{\star} \cdot a \rangle tt$, $\phi_{of}(a) = [\overline{\mathcal{B}}^{\star} \cdot a]f\!f$ and $\alpha_{el}(a) = \{a\}$ for all $a \in \overline{\mathcal{B}}$ if, and only if, it does not admit $\mathcal{B}$-progressing paths that satisfy weak $\mathcal{B}$-hyperfairness of actions and are $(\rho, \alpha_f, \alpha_e)$-violating.*

Proof. We argue this assignment to ϕ_{on}, ϕ_{of} and α_{el} satisfies the four properties of finitely realisable predicates.

1. The invariant property: let π be an arbitrary path. According to $\mathcal{B}$-WHFA, π is complete if for all suffixes of π, all non-blocking actions that are perpetually $\mathcal{B}$-reachable on the suffix also occur in that suffix. We prove this is equivalent to the condition that a path π is complete if for all states s in π and all non-blocking actions a, if s satisfies $\langle \overline{\mathcal{B}}^{\star} \cdot a \rangle tt$ then the suffix of π starting in s contains a state satisfying $[\overline{\mathcal{B}}^{\star} \cdot a]f\!f$ or an occurrence of a. Refer to this condition as P. Let π' be an arbitrary suffix of π and let s be the first state of π'. Let a be an arbitrary non-blocking action. If a is not $\mathcal{B}$-reachable from s, then WHFA does not require a to occur in π' since it is not perpetually reachable, and P does not require a to occur in π' because $\langle \overline{\mathcal{B}}^{\star} \cdot a \rangle tt$ is not satisfied in s. If a is $\mathcal{B}$-reachable from s, we consider if a is perpetually $\mathcal{B}$-reachable on π'. If it is not, WHFA does not require a to occur in π'. Also P does not require this, since within finitely many steps a state will occur on π' in which $[\overline{\mathcal{B}}^{\star} \cdot a]f\!f$ is satisfied. If a is perpetually $\mathcal{B}$-reachable in π', then WHFA requires a to occur on π'. Since there are no states on π' that satisfy $[\overline{\mathcal{B}}^{\star} \cdot a]f\!f$, P also requires an occurrence of a in π'. Hence, in all cases, WHFA and P give the exact same requirements on action occurrences in π'.

2. The locking property: if a state is $\mathcal{B}$-locked, no non-blocking actions are enabled in that state. In that case, no non-blocking actions can be $\mathcal{B}$-reachable since there must be an occurrence of a blocking action before a state can be reached in which a non-blocking action is enabled. In the other direction, if a state has no $\mathcal{B}$-reachable non-blocking actions, then no non-blocking actions are enabled since every enabled action is also reachable. Hence, the two conditions coincide.

3. The exclusive property: $\langle \overline{\mathcal{B}}^{\star} \cdot a \rangle tt = \neg[\overline{\mathcal{B}}^{\star} \cdot a]f\!f$ for all $a \in Act$, so this follows directly.

4. The persistent property: this follows from $\langle \overline{\mathcal{B}}^\star \cdot a \rangle tt = \neg [\overline{\mathcal{B}}^\star \cdot a] ff$ as well. If there is no state on a path that satisfies $[\overline{\mathcal{B}}^\star \cdot a] ff$ for some $a \in \overline{\mathcal{B}}$, then the final state of that path satisfies $\neg [\overline{\mathcal{B}}^\star \cdot a] ff$ and hence satisfies $\langle \overline{\mathcal{B}}^\star \cdot a \rangle tt$.

Hence, $\mathcal{B}$-WHFA is a finitely realisable predicate on paths.

That $\mathcal{B}$-WHFA is feasible was proven as Proposition 37. We apply Theorem 24 to conclude the theorem holds. ◄

D.3.3 JA

Finally, we prove Corollary 27.

▶ **Corollary 27.** *A state in an LTSC satisfies Formula 2 with $\phi_{on}(a) = \langle a \rangle tt$, $\phi_{of}(a) = ff$ and $\alpha_{el}(a) = \{b \in Act \mid a \not\mathrel{\smash{\nearrow}}\!\!\bullet\, b\}$ for all $a \in \overline{\mathcal{B}}$ if, and only if, it does not admit $\mathcal{B}$-progressing paths that satisfy $\mathcal{B}$-justness of actions and are $(\rho, \alpha_f, \alpha_e)$-violating.*

Proof. We use $\#a$ as shorthand for $\{b \in Act \mid a \not\mathrel{\smash{\nearrow}}\!\!\bullet\, b\}$ from here on. We argue this assignment to ϕ_{on}, ϕ_{of} and α_{el} satisfies the four properties of finitely realisable predicates.

1. The invariant property: we need to show that an arbitrary path π satisfies JA exactly when it satisfies the condition that for all states s of π and all $a \in \overline{\mathcal{B}}$, if s satisfies $\langle a \rangle tt$ then in the suffix of π starting in s, there must be a state that satisfies ff or an occurrence of an action in $\#a$. We refer to this latter condition as P. Justness requires that for every state in π, every action that is enabled must be eliminated in the suffix of π starting in s. The correspondence between P and JA follows directly from the observation that there can never be a state that satisfies ff, hence P also requires that after a non-blocking action is enabled, it must subsequently be eliminated.
2. The locking property: if a state is $\mathcal{B}$-locked, no non-blocking actions are enabled in that state. This is identical to the definition that $\langle a \rangle tt$ is violated for all $a \in \overline{\mathcal{B}}$.
3. The exclusive property: ff is violated in every state, hence this property is trivially satisfied.
4. The persistent property: this follows directly from the second requirement on concurrency relations on actions.

Hence, $\mathcal{B}$-JA is a finitely realisable predicate on paths.

That $\mathcal{B}$-JA is feasible was proven as Proposition 39. We apply Theorem 24 to conclude the theorem holds. ◄

D.4 Proof of SFA and SHFA Formulae

We here prove Theorem 28 and Theorem 29. We prove both by first proving a more generic theorem. For this, we introduce an assumption that generalises both strong fairness and strong hyperfairness, which we name P_S. P_S depends on a mapping ϕ_{of} from non-blocking actions to closed modal μ-calculus formulae. The assumption P_S is then as follows: a path π satisfies P_S if for every non-blocking action a , it holds that a occurs infinitely often on π or there is a suffix of π on which $\phi_{of}(a)$ is perpetually satisfied. In line with previous definitions, "perpetually satisfied" means satisfied in every state.

▶ **Theorem 56.** *An LTS satisfies Formula 8 if, and only if, its initial state does not admit a $\mathcal{B}$-progressing path that is $(\rho, \alpha_f, \alpha_e)$-violating, and satisfies P_S, for a given choice of ϕ_{of} such that: P_S is feasible, and a state is $\mathcal{B}$-locked if, and only if, it satisfies $\phi_{of}(b)$ for all $b \in \overline{\mathcal{B}}$.*

Here Formula 8 is as follows:

$$\neg\langle\rho\cdot\overline{\alpha_f}^\star\rangle(\langle\alpha_e\rangle tt \vee [\overline{\mathcal{B}}]ff \vee \bigvee_{\emptyset\neq F\subseteq\overline{\mathcal{B}}} \nu X.(\bigwedge_{a\in F}\mu W.((\bigwedge_{b\in\overline{\mathcal{B}}\setminus F}\phi_{of}(b))\wedge(\langle a\setminus\alpha_f\rangle X \vee \langle\overline{\alpha_f}\rangle W)))) \quad (8)$$

We once again fix $\mathcal{B},\rho,\alpha_f$ en α_e. We also fix an arbitrary mapping ϕ_{of} such that P_S is feasible, and a state is $\mathcal{B}$-locked if, and only if, it satisfies $\phi_{of}(b)$ for all $b\in\overline{\mathcal{B}}$. We first have to prove some supporting propositions.

▶ **Proposition 57.** *Every path π that satisfies P_S is also $\mathcal{B}$-progressing*

Proof. Let π be an arbitrary path. If π is infinite, it is trivially $\mathcal{B}$-progressing. We hence assume it is finite. Towards a contradiction, assume π is not $\mathcal{B}$-progressing. Let s be the final state of π, it must by our assumption not be a $\mathcal{B}$-locked state. Then by our restriction on ϕ_{of}, there must exist an action $b\in\overline{\mathcal{B}}$ such that s does not satisfy $\phi_{of}(b)$. Since π is finite, no actions can occur infinitely often, so b does not occur infinitely often. Since π satisfies P_S, it must be the case that there is a suffix of π on which $\phi_{of}(b)$ is perpetually satisfied. But it is not satisfied in the final state of π, which is in every suffix of π. Hence, we have a contradiction and we conclude that s is a $\mathcal{B}$-locked state, and so π satisfies $\mathcal{B}$-progress. ◀

▶ **Proposition 58.** *Every $\mathcal{B}$-progressing and finite path satisfies P_S.*

Proof. Let π be an arbitrary $\mathcal{B}$-progressing and finite path. Since π is finite and $\mathcal{B}$-progressing, it has a final state s and all actions enabled in s are blocking. Since s is a $\mathcal{B}$-locked state, by our choice of ϕ_{of} we know $\phi_{of}(b)$ is satisfied in s for all $b\in\overline{\mathcal{B}}$. The suffix of π consisting only of s is a suffix in which $\phi_{of}(b)$ is perpetually satisfied for all $b\in\overline{\mathcal{B}}$, so π satisfies P_S. ◀

▶ **Proposition 59.** *Let π be an path that satisfies P_S, then every path π' of which π is a suffix also satisfies P_S.*

Proof. Let π' be an arbitrary path which contains a suffix π such that π satisfies P_S. We prove π' satisfies P_S as well. To this end, we prove that every non-blocking action $b\in\overline{\mathcal{B}}$ occurs infinitely often in π' or π' has a suffix on which $\phi_{of}(b)$ is perpetually satisfied. Let b be an arbitrary non-blocking action in $\overline{\mathcal{B}}$. Then since π satisfies P_S, b occurs infinitely often in π or π has a suffix on which $\phi_{of}(b)$ is perpetually satisfied. Both qualities are inherited by any path of which π is a suffix, and hence π' also satisfies at least one of the two conditions for b. We conclude that π' satisfies P_S. ◀

▶ **Proposition 60.** *Let π be path that satisfies P_S, then every suffix π' of π also satisfies P_S.*

Proof. Let π be an arbitrary path that satisfies P_S and let π' be an arbitrary suffix of that path. We prove π' satisfies P_S. To this end, let b be an arbitrary non-blocking action in $\overline{\mathcal{B}}$. We show that b occurs infinitely often in π' or that π' has a suffix on which $\phi_{of}(b)$ is perpetually satisfied. Since π satisfies P_S, we know that at least one of the two cases is true for π.

- If b occurs infinitely often in π, then it also occurs infinitely often in π'.
- If π has a suffix π'' on which $\phi_{of}(b)$ is perpetually satisfied, then π'' is a suffix of π' as well, or a suffix of π'' is also a suffix of π'. Either way, π' also has a suffix on which $\phi_{of}(b)$ is perpetually satisfied.

We conclude that π' satisfies P_S. ◀

We now prove the correctness of the formula. We first split the formula into multiple sub-formulae.

$$violate_S = \langle \rho \rangle finite_S$$
$$finite_S = \langle \overline{\alpha_f}^\star \rangle (\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}]ff \vee disjunct_S)$$
$$disjuct_S = \bigvee_{\emptyset \neq F \subseteq \overline{\mathcal{B}}} invariant_S(F)$$
$$invariant_S(F) = \nu X.(\bigwedge_{a \in F} eliminate_S(F, a))$$
$$eliminate_S(F, a) = \mu W.((\bigwedge_{b \in \overline{\mathcal{B}} \setminus F} \phi_{of}(b)) \wedge (\langle a \setminus \alpha_f \rangle X \vee \langle \overline{\alpha_f} \rangle W))$$

We have that Formula 8 $= \neg violate_S$.

We characterise the semantics of the formula bottom-up, starting with $eliminate_S$.

▶ **Lemma 61.** *For all environments e, states $s \in \mathcal{S}$, sets $F \subseteq \overline{\mathcal{B}}, \mathcal{F} \subseteq \mathcal{S}$ and non-blocking actions $a \in F$, it is the case that $s \in [\![eliminate_S(F, a)]\!]_{e[X:=\mathcal{F}]}$ if, and only if, s admits a finite path π with final state s_{final} satisfying the following requirements:*
1. *π is α_f-free, and*
2. *all states of π, with the possible exception of s_{final}, satisfy $\phi_{of}(b)$ for all $b \in \overline{\mathcal{B}} \setminus F$, and*
3. *s_{final} is in $\mathcal{F}$, and*
4. *the final transition of π, t_{final} is labelled with a.*

Proof. Let s be an arbitrary state, $F \subseteq \overline{\mathcal{B}}$ and $\mathcal{F} \subseteq \mathcal{S}$ arbitrary sets and a an arbitrary action in F, a is by definition of F non-blocking. We prove s is in the semantics of $eliminate_S(F, a)$ under environment $e[X := \mathcal{F}]$ if, and only if, it admits a path π meeting conditions 1 through 4.

Note that $\phi_{of}(b)$ does not depend on Y regardless of the choice of b, since we require ϕ_{of} to map to closed modal μ-calculus formulae. It is also that case that $\langle a \setminus \alpha_f \rangle X$ does not depend on Y. Hence, we can apply Proposition 40 to conclude that $s \in [\![eliminate_S(F, a)]\!]_{e[X:=\mathcal{F}]}$ if, and only if, s admits a finite path π' meeting the following conditions:
5. all actions occurring in π' are in $\overline{\alpha_f}$, and
6. all states in π' satisfy $\left[\!\!\left[\bigwedge_{b \in \overline{\mathcal{B}} \setminus F} \phi_{of}(b)\right]\!\!\right]_{e[X:=\mathcal{F}]}$, and
7. the final state of π' is in $[\![\langle a \setminus \alpha_f \rangle X]\!]_{e[X:=\mathcal{F}]}$.

We prove the claim by showing that if s admits a path π satisfying conditions 1 through 4, it also admits a path π' satisfying conditions 5 through 7, and vice versa.

- If s admits a path π satisfying conditions 1 through 4, then we know π's last transition is labelled with a. Let π' be the prefix of π missing only the last transition and final state. Then because π is α_f-free (by 1), so is π' (satisfying 5). Additionally, because all states on π with the possible exception of s_{final} are in $\left[\!\!\left[\bigwedge_{b \in \overline{\mathcal{B}} \setminus F} \phi_{of}(b)\right]\!\!\right]_{e[X:=\mathcal{F}]}$ (by 2), and π' has all states of π except s_{final}, all states in π' are in this set as well (satisfying 6). Finally, the final state of π', s'_{final}, admits a transition labelled with a to s_{final}, namely t_{final} (by 4), and $s_{final} \in \mathcal{F}$ (by 3). Hence, $s'_{final} \in [\![\langle a \rangle X]\!]_{e[X:=\mathcal{F}]}$. Since a occurs in π (by 4) and π is α_f-free (by 1), we know $a \notin \alpha_f$. Therefore, $s'_{final} \in [\![\langle a \setminus \alpha_f \rangle X]\!]_{e[X:=\mathcal{F}]}$ (satisfying 7). We have shown s admits a path π' satisfying conditions 5 through 7.
- If s admits a path π' satisfying conditions 5 through 7, then we know that the final state of π', s'_{final}, admits an a transition to a state in $\mathcal{F}$, and that $a \notin \alpha_f$ (by 7). Let t_{final} be

such a transition, and let s_{final} be the target of t_{final}. We take $\pi = \pi' t_{final} s_{final}$. Since π' is α_f-free (by 5) and the only new transition is also not labelled with an action in α_f, π is α_f-free (satisfying 1). Additionally, all states on π' satisfy $\phi_{of}(b)$ for all $b \in \overline{\mathcal{B}} \setminus F$ (by 6) and so this also holds for all states on π with the possible exception of s_{final} (satisfying 2). Finally, by construction $s_{final} \in \mathcal{F}$ and $act(t_{final}) = a$ (satisfying 3 and 4). We conclude that s admits a path π meeting conditions 1 through 4.

By Proposition 40, we have proven that $s \in [\![eliminate_S(F, a)]\!]_{e[X := \mathcal{F}]}$ if, and only if, s admits a path satisfying conditions 1 through 4.   ◄

Next, we want to establish the semantics of $invariant_S(F)$. We here fix an arbitrary $F \subseteq \overline{\mathcal{B}}$ such that F is not the empty set. Let $S_{inf(F)}$ be the set of all states that admit infinite, α_f-free paths along which all actions in F occur infinitely often and $\phi_{of}(b)$ is perpetually satisfied for all $b \in \overline{\mathcal{B}} \setminus F$. We prove that $S_{inf(F)}$ is a fixed point of the transformer characterised by $invariant_S(F)$. We will subsequently prove that it is the greatest fixed point.

▶ **Lemma 62.** *$S_{inf(F)}$ is a fixed point of the transformer $T_{inf(F)}$ defined by:*

$$T_{inf(F)}(\mathcal{F}) = \bigcap_{a \in F} \left\{ s \in \mathcal{S} \mid s \in [\![eliminate_S(F, a)]\!]_{e[X := \mathcal{F}]} \right\}$$

for arbitrary environment e.

Proof. We prove $S_{inf(F)}$ is a fixed point of $T_{inf(F)}$ through mutual set inclusion of $S_{inf(F)}$ and $T_{inf(F)}(S_{inf(F)})$.

▪ Let s be an arbitrary state in $S_{inf(F)}$, we prove that $s \in T_{inf(F)}(S_{inf(F)})$. Since $s \in S_{inf(F)}$, we know that s admits an infinite, α_f-free path π such that all actions in F occur infinitely often in π and $\phi_{of}(b)$ is perpetually satisfied in π for all $b \in \overline{\mathcal{B}} \setminus F$. To prove $s \in T_{inf(F)}(S_{inf(F)})$, we need to show that for all $a \in F$, it is the case that $s \in [\![eliminate_S(F, a)]\!]_{e[X := S_{inf(F)}]}$. To this end, let a be an arbitrary action in F. We now such an action exists since we assumed $F \neq \emptyset$. We prove $s \in [\![eliminate_S(F, a)]\!]_{e[X := S_{inf(F)}]}$. By Lemma 61, it suffices to prove that s admits a finite path π' with final state s'_{final} and final transition t'_{final} such that π' is α_f-free, $\phi_{of}(b)$ is satisfied for all $b \in \overline{\mathcal{B}} \setminus F$ in all states of π' (with the possible exception of s'_{final}), $s'_{final} \in S_{inf(S)}$ and $act(t'_{final}) = a$. Since $a \in F$, a is non-blocking and occurs infinitely often in π. Let t'_{final} be the first transition of π labelled with a, and let $s'_{final} = trgt(t'_{final})$. Let π' be the prefix of π ending in s'_{final}. Note that since π is α_f-free, so is π'. Additionally, since $\phi_{of}(b)$ is satisfied for all $b \in \overline{\mathcal{B}} \setminus F$ in all states of π, this holds for all states of π' as well. By construction, $act(t'_{final}) = a$. It remains to prove that $s'_{final} \in S_{inf(F)}$. Let π'' be the suffix of π starting in s'_{final}. This suffix inherits many properties from π, particularly that it is infinite, α_f-free, that all actions in F occur infinitely often and that $\phi_{of}(b)$ is perpetually satisfied for all non-blocking actions not in F. Hence, π'' witnesses $s'_{final} \in S_{inf(F)}$. We conclude that $s \in [\![eliminate_S(F, a)]\!]_{e[X := S_{inf(F)}]}$ for all $a \in F$ and hence $s \in T_{inf(F)}(S_{inf(F)})$.

▪ Let s be an arbitrary state in $T_{inf(F)}(S_{inf(F)})$, we prove $s \in S_{inf(F)}$. From $s \in T_{inf(F)}(S_{inf(F)})$, we conclude that for all $a \in F$, $s \in [\![eliminate_S(F, a)]\!]_{e[X := S_{inf(F)}]}$. To prove $s \in S_{inf(F)}$, we construct a path π from s that is infinite, α_f-free, and on which all actions in F occur infinitely often and $\phi_{of}(b)$ is perpetually satisfied for all $b \in \overline{\mathcal{B}} \setminus F$. To this end, let a be an arbitrary action in F, which is guaranteed to exist because $F \neq \emptyset$. We will use $s \in [\![eliminate_S(F, a)]\!]_{e[X := S_{inf(F)}]}$ to construct π. By Lemma 61, s admits a finite path π' with a final state s'_{final} and a final transition t'_{final} such that π' is α_f-free, ϕ_{of} is satisfied for all non-blocking actions not in F in all states of π' with the possible

exception of s'_{final}, $s'_{final} \in S_{inf(F)}$ and $act(t'_{final}) = a$. From $s'_{final} \in S_{inf(F)}$, we know that s'_{final} admits an infinite, α_f-free path π'' such that all actions in F occur infinitely often in π'' and ϕ_{of} is perpetually satisfied for all non-blocking actions not in F in π''. Note that since s'_{final} is on π'', ϕ_{of} is satisfied for all non-blocking actions not in F in s'_{final}, even though we did not get this fact directly from Lemma 61. Let $\pi = \pi' \cdot \pi''$. Since π'' is infinite and all actions in F occur infinitely often in π'', both qualities hold for π as well. Additionally, since both π' and π'' are α_f-free, so is π. It remains to show that ϕ_{of} is perpetually satisfied for all non-blocking actions not in F on π. We got from Lemma 61 that all states on π' satisfy this condition, except possibly s'_{final}. But from $s'_{final} \in S_{inf(F)}$, we got that this holds for s'_{final} as well. It also holds for all states on π''. Hence, the condition is satisfied for all states of π. Using π as a witness, we conclude $s \in S_{inf(F)}$.

We conclude that $S_{inf(F)} = T_{inf(F)}(S_{inf(F)})$.   ◄

▶ **Lemma 63.** *The set $S_{inf(F)}$ is the greatest fixed point of the transformer defined in Lemma 62.*

Proof. For this proof, we use Lemma 62. Let $\mathcal{F}$ be an arbitrary fixed point of $T_{inf(F)}$, we prove $\mathcal{F} \subseteq S_{inf(F)}$. To this end, let s be an arbitrary state in $\mathcal{F}$, we prove $s \in S_{inf(F)}$ by constructing a path π from s that is infinite, α_f-free, on which all actions in F occur infinitely often and $\phi_{of}(b)$ is perpetually satisfied for all $b \in \overline{\mathcal{B}} \setminus F$.

For this construction, let Q be a queue of actions. This construction consists of an infinite loop. We start the loop with $\pi = s$ and Q containing exactly one copy of every action in F in arbitrary order. Note that $s \in \mathcal{F}$. The loop as several invariants

- the final state of π is in $\mathcal{F}$.
- Q contains exactly one copy of every action in F and nothing else.
- π constructed so far is α_f-free.

In every iteration of the loop, let s' be the final state of π constructed so far. Let a be the action at the head of Q. Since $F \neq \emptyset$, and Q always contains a copy of every action in F, such an a always exists. We pop a from Q and immediately re-insert it at the end of the queue. An invariant of this construction is that s' is always in $\mathcal{F}$. Because of this, and since $\mathcal{F}$ is a fixed point of $T_{inf(F)}$, we have that $s' \in T_{inf(F)}(\mathcal{F})$. From this we conclude that for all actions $a' \in F$, $s' \in [\![eliminate_S(F, a')]\!]_{e[X:=\mathcal{F}]}$. The action a came from Q, and hence is an action in F. Therefore, $s' \in [\![eliminate_S(F, a)]\!]_{e[X:=\mathcal{F}]}$. By Lemma 61, s' therefore admits a path π' meeting the following conditions:

1. π' is α_f-free, and
2. all states of π', with the possible exception of its final state, satisfy $\phi_{of}(b)$ for all $b \in \overline{\mathcal{B}} \setminus F$, and
3. the final state of π' is in $\mathcal{F}$, and
4. the final transition of π' is labelled with a.

We append π' to π, and then start the loop again. Note that the new final state of π is in $\mathcal{F}$ by construction, and we the only modification of Q is removing and then re-inserting a, so Q still contains exactly one copy of every action in F and nothing else. Since we added a α_f-free segment to π, π is still α_f-free. Hence, all the invariants still hold.

This loop will never terminate. From this, and the invariants, it immediately follows that the path π we construct is infinite and α_f-free. It remains to argue that all actions in F occur infinitely often in π and that ϕ_{of} is perpetually satisfied for all non-blocking actions not in F. Since Q is a queue, and we always pop the top action and then insert it back at the end, and $F \subseteq \overline{\mathcal{B}} \subseteq Act$ and Act is finite, the construction of π ensures that every action

that is initially in Q is selected infinitely often to be a. We then add a segment to π where a occurs. Hence, every action that is initially in Q occurs infinitely often in π. We initialised Q with all the actions in F, hence every action in F occurs infinitely often in π. At first glance, it may seem that this construction does not guarantee the second condition. After all, every path segment we add only ensures that all states except the last state satisfy $\phi_{of}(b)$ for all $b \in \overline{\mathcal{B}} \setminus F$. Note however, that these segments do guarantee that the final state is in the fixed point $\mathcal{F}$ again, and therefore that the final state once again admits a segment that meets the condition that all states, with the possible exception of the last, satisfy this condition. Additionally, the added path always contains at least one transition, so the initial state of the segment is not the final state. Hence, whenever we add a segment π', the fact that we can add a next segment guarantees that ϕ_{of} is satisfied for all non-blocking actions not in F in the final state of π'. And since we infinitely often can add a next segment, we conclude that every state on π satisfies the condition.

We have constructed a path π witnessing $s \in S_{inf(F)}$, hence $S_{inf(F)}$ is the greatest fixed point of $T_{inf(F)}$. ◀

Since the semantics of $invariant_S(F)$ are exactly the greatest fixed point of $T_{inf(F)}$, we can conclude the following.

▶ **Corollary 64.** *The set of states characterised by $invariant_S(F)$ is exactly the set of states that admit infinite, α_f-free paths on which all actions in F occur infinitely often and $\phi_{of}(b)$ is perpetually satisfied for all $b \in \overline{\mathcal{B}} \setminus F$. This holds true whenever $F \subseteq \overline{\mathcal{B}}$ and $F \neq \emptyset$.*

We now cover the semantics of $disjunct_S$.

▶ **Lemma 65.** *For all environments e and $s \in \mathcal{S}$, $s \in [\![disjunct_S]\!]_e$ if, and only if, there exists a subset F of $\overline{\mathcal{B}}$ with $F \neq \emptyset$, such that s admits an infinite, α_f-free path on which all actions in F occur infinitely often and $\phi_{of}(b)$ is perpetually satisfied for all $b \in \overline{\mathcal{B}} \setminus F$.*

Proof. This follows trivially from the definition of $disjunct_S$, Corollary 64 and the definition of the modal μ-calculus semantics. ◀

We next cover $finite_S$.

▶ **Lemma 66.** *For all environments e and states $s \in \mathcal{S}$, $s \in [\![finite_S]\!]_e$ if, and only if, s admits a complete path π that is $(\varepsilon, \alpha_f, \alpha_e)$-violating and satisfied P_S.*

Proof. Note that $finite_S$ is equivalent to $\mu Y.(\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}]ff \vee disjunct_S \vee \langle \overline{\alpha_f} \rangle Y)$, we merely wrote it more compactly using regular formulae.

We wish to apply Proposition 40 to determine the semantics of this formula. To be able to do this, we observe that $\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}]ff \vee disjunct_S$ does not depend on Y. Hence, we can apply Proposition 40 (with $\phi_1 = tt$) to conclude that s is in $[\![finite_S]\!]_e$ if, and only if, s admits a finite path π' that is α_f-free and ends in a state satisfying $[\![\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}]ff \vee disjunct_S]\!]_e$. We use this information to prove the bi-implication.

- Assume $s \in [\![finite_S]\!]_e$, then by Proposition 40, we obtain a path π' starting in s that is α_f-free and ends in a state s' such that $s' \in [\![\langle \alpha_e \rangle tt]\!]_e$, or $s' \in [\![[\overline{\mathcal{B}}]ff]\!]_e$, or $s' \in [\![disjunct_S]\!]_e$. We construct the path π using this information. For this, we do a case distinction on which set s' is in.
  - If $s' \in [\![\langle \alpha_e \rangle tt]\!]_e$, then π' is a finite path with an action in α_e enabled in the final state. We extend π' with a transition labelled with an action in α_e. The resulting path can be extended to a path π that satisfies P_S because we assumed P_S to be feasible and

we know this path is also $\mathcal{B}$-progressing by Proposition 57. Note that π' is α_f-free, and we immediately appended a transition labelled with an action in α_e to it, before we did the extension to a $\mathcal{B}$-progressing and complete path. Hence, the extended path π is α_f-free up until an occurrence of α_e, and hence is also α_f-free up until the first occurrence of such an action. We conclude that π is $\mathcal{B}$-progressing, α_f-free up until the first occurrence of an action in α_e, and satisfies P_S.

- If $s' \in [\![\overline{\mathcal{B}}]f\!f]\!]_e$, then π' is a $\mathcal{B}$-progressing path ending in a blocking state. Since π' is finite and $\mathcal{B}$-progressing, it satisfies P_S according to Proposition 58. Additionally, we know that π' is entirely α_f-free. We conclude that s admits a path that is $\mathcal{B}$-progressing, entirely α_f-free and satisfies P_S.

- If $s' \in [\![disjunct_S]\!]_e$ then by Lemma 65, there exists a subset F of $\overline{\mathcal{B}}$ with $F \neq \emptyset$ such that s' admits an infinite, α_f-free path π'' on which all actions in F occur infinitely often and $\phi_{of}(b)$ is perpetually satisfied for all $b \in \overline{\mathcal{B}} \setminus F$. Let F be a subset meeting this condition, and let π'' be the path we obtain from F. Then, let $\pi = \pi' \cdot \pi''$. Since π'' is infinite, so is π, hence π is $\mathcal{B}$-progressing. Since both π' and π'' are α_f-free, so is π. It remains to show that π satisfies P_S.

  To do this, we must show that for every $a \in \overline{\mathcal{B}}$, a occurs infinitely often in π or π has a suffix on which $\phi_{of}(a)$ is perpetually satisfied. Let a be an arbitrary non-blocking action. If $a \in F$, then since all actions in F occur infinitely often on π'', a occurs infinitely often on π. If $a \notin F$, then $a \in \overline{\mathcal{B}} \setminus F$ and hence $\phi_{of}(a)$ is perpetually satisfied on π''. Therefore, π contains a suffix on which $\phi_{of}(a)$ is perpetually satisfied.

  We have constructed a path π that is $\mathcal{B}$-progressing, α_f-free and satisfying P_S.

  In all three cases, we can construct a $\mathcal{B}$-progressing path that is $(\varepsilon, \alpha_f, \alpha_e)$-violating and satisfies P_S.

- Assume that s admits a path π that is $\mathcal{B}$-progressing, $(\varepsilon, \alpha_f, \alpha_e)$-violating and satisfying P_S. We prove $s \in [\![finite_S]\!]_e$. By Proposition 40, it is sufficient to show that s admits a finite path π' that is α_f-free and ends in a state s' that is in $[\![\langle\alpha_e\rangle tt]\!]_e$, or in $[\![\overline{\mathcal{B}}]f\!f]\!]_e$, or in $[\![disjunct_S]\!]_e$. We do a case distinction on whether there are any occurrences of actions in α_e in π.

  - If there are, then let t_{en} be the first transition in π that is labelled with an action in α_e. Let π' be the prefix of π ending in $src(t_{en})$. Since π is α_f-free up until the first occurrence of an action in α_e, π' is α_f-free. Additionally, because t_{en} is enabled in the final state of π', this state is in $[\![\langle\alpha_e\rangle tt]\!]_e$. We can therefore conclude $s \in [\![finite_S]\!]_e$.

  - If there are not, then π is entirely α_f-free. We do a second case distinction on whether π is finite or not.

    * If π is finite, then the final state of π is a $\mathcal{B}$-locked state. Hence, π is an entirely α_f-free path ending in a state in $[\![\overline{\mathcal{B}}]f\!f]\!]_e$. We conclude that $s \in [\![finite_S]\!]_e$.

    * If π is infinite we need a more extensive argument. Recall that π satisfies P_S. Hence all non-blocking actions a occur infinitely often on π or there is a suffix of π on which $\phi_{of}(a)$ is perpetually satisfied. Let F be the set of all non-blocking actions that occur infinitely often on π. If $F = \emptyset$, then it must be the case that π has a suffix π'' on which $\phi_{of}(a)$ is perpetually satisfied for all $a \in \overline{\mathcal{B}}$. Consider that we assumed that P_S is such that a state is $\mathcal{B}$-locked if, and only if, it satisfies $\phi_{of}(b)$ for all $b \in \overline{\mathcal{B}}$. Therefore, every state on π'' is a $\mathcal{B}$-locked state. Any such state is in $[\![\overline{\mathcal{B}}]f\!f]\!]_e$. Let π' be the finite prefix of π ending in the first state of π'', then π' is α_f-free because π is and ends in a state in $[\![\overline{\mathcal{B}}]f\!f]\!]_e$. Since s admits π', $s \in [\![finite_S]\!]_e$. If $F \neq \emptyset$, then s admits a path π that is infinite, α_f-free and on which all actions in F occur infinitely often and that has a suffix, π'' on which $\phi_{of}(b)$ is perpetually

satisfied for all $b \in \overline{\mathcal{B}} \setminus F$. Let s' be the first state of π''. Since s' admits π'', it admits an infinite, α_f-free path on which all actions in F occur infinitely often and $\phi_{of}(b)$ is perpetually satisfied for all $b \in \overline{\mathcal{B}} \setminus F$. Hence, by Lemma 65, $s' \in [\![disjunct_S]\!]_e$. Let π' be the path such that $\pi = \pi' \cdot \pi''$. Then π' is a finite path from s that is α_f-free and which ends in a state that is in $[\![disjunct_S]\!]_e$. Therefore, $s \in [\![finite_S]\!]_e$. We have proven that in all cases, $s \in [\![finite_S]\!]_e$.

We have proven both sides of the bi-implication. ◀

The final two steps of the general proof are largely trivial.

▶ **Lemma 67.** *For all environments e and states $s \in \mathcal{S}$, it holds that $s \in [\![violate_S]\!]_e$ if, and only if, s admits a path that is $\mathcal{B}$-progressing, satisfies P_S and is $(\rho, \alpha_f, \alpha_e)$-violating.*

Proof. It follows directly from the definition of the diamond operator that $violate_S$ characterises those states that admit a path π that has a prefix matching ρ that ends in some state s' in $[\![finite_S]\!]_e$. By Lemma 66, s' admits a $\mathcal{B}$-progressing path π' that satisfies P_S and is $(\varepsilon, \alpha_f, \alpha_e)$-violating. By Proposition 59, prepending π to π' results in a path that still satisfies P_S. It is also still $\mathcal{B}$-progressing, and by construction, it has a prefix matching ρ, namely π, after which it is α_f-free up until the first occurrence of an action in α_e. Hence, it is $(\rho, \alpha_f, \alpha_e)$-violating. ◀

Formula 8 is the negation of $violate_S$, hence it expresses that a state does not admit such a path. Theorem 56 follows directly.

To be able to apply Theorem 56 to SFA and SHFA, we need to prove the following: firstly, that both assumptions are feasible, this has already been proven as Proposition 35 and Proposition 38. Secondly, that there exist choices of ϕ_{of} such that a state is $\mathcal{B}$-locked if, and only if, ϕ_{of} is satisfied for all non-blocking actions and such that the resulting P_S is equivalent to SFA or SHFA respectively.

D.4.1   SFA

We first prove Theorem 28.

▶ **Theorem 28.** *An LTS satisfies Formula 3 if, and only if, its initial state does not admit $\mathcal{B}$-progressing paths that satisfy $\mathcal{B}$-strong fairness of actions and are $(\rho, \alpha_f, \alpha_e)$-violating.*

Where Formula 3 is:

$$\neg \langle \rho \cdot \overline{\alpha_f}^\star \rangle (\langle \alpha_e \rangle tt \vee [\overline{\mathcal{B}}] ff \vee \bigvee_{\emptyset \neq F \subseteq \overline{\mathcal{B}}} \nu X.(\bigwedge_{a \in F} \mu W.((\bigwedge_{b \in \overline{\mathcal{B}} \setminus F} [b] ff) \wedge (\langle a \setminus \alpha_f \rangle X \vee \langle \overline{\alpha_f} \rangle W))))$$

Proof. To get Formula 3 from Formula 8, we take $\phi_{of}(b) = [b] ff$ for all $b \in \overline{\mathcal{B}}$. This is a mapping to closed modal μ-calculus formulae, and hence a valid choice of ϕ_{of}. By definition of a $\mathcal{B}$-locked state, every $\mathcal{B}$-locked state satisfies $\phi_{of}(b)$ for all $b \in \overline{\mathcal{B}}$, and vice versa. We show that a path π satisfies $\mathcal{B}$-strong fairness of actions (Definition 12) exactly when it satisfies P_S interpreted on this ϕ_{of}. Note that in both theorems, it is included that the path must be $\mathcal{B}$-progressing so we take that as given. A path π satisfies $\mathcal{B}$-SFA if, and only if, for every suffix π' of π, every action $a \in \overline{\mathcal{B}}$ that is relentlessly enabled in π' occurs in π'.

First note that if an action a is relentlessly enabled on a suffix π' of π, then a is also relentlessly enabled on π. After all, every suffix of π is either also a suffix of π' or contains π', and in either case contains a state in which a is enabled, since every suffix of π' contains such

a state. Similarly, if an action a is relentlessly enabled on π, it is also relentlessly enabled on an arbitrary suffix π' of π, since every suffix of π' is also a suffix of π. Therefore, $\mathcal{B}$-SFA can be rewritten as follows: a path π satisfies $\mathcal{B}$-SFA if, and only if, every action $a \in \overline{\mathcal{B}}$ that is relentlessly enabled in π occurs in every suffix π' of π.

If a path is $\mathcal{B}$-progressing and finite, then no non-blocking action can be relentlessly enabled on that path, since no non-blocking actions are enabled in the final state. If a path is infinite, then if an action occurs in every suffix of the path then it must occur infinitely often. Therefore, $\mathcal{B}$-SFA can be rewritten as follows: a path π satisfies $\mathcal{B}$-SFA if, and only if, every action $a \in \overline{\mathcal{B}}$ that is relentlessly enabled in π occurs infinitely often on π.

We need to make one last step to connect $\mathcal{B}$-SFA to P_S: if an action is not relentlessly enabled on some path, then the path must contain a suffix on which that action is never enabled. Hence, every non-blocking action is either relentlessly enabled on some path, or the path has a suffix on which the action is perpetually disabled. In other words, a path π satisfies $\mathcal{B}$-SFA if, and only if, every action $a \in \overline{\mathcal{B}}$ either occurs infinitely often on π or π has a suffix on which a is perpetually disabled. This is almost exactly the definition of P_S when $\phi_{of}(b) = [b]\mathit{ff}$ for all $b \in \overline{\mathcal{B}}$, with the only exception being that $\mathcal{B}$-SFA has an exclusive or, and P_S has a non-exclusive or. However, if an action occurs infinitely often then it cannot become perpetually disabled, and vice-versa. So in this case, the choice of ϕ_{of} enforces an exclusive or even though this is not part of the definition of P_S. Hence, the two definitions coincide.

We have shown that P_S when taking $\phi_{of}(b) = [b]\mathit{ff}$ for all $b \in \overline{\mathcal{B}}$ is equivalent to $\mathcal{B}$-strong fairness of actions. To be able to apply Theorem 56 to Formula 3, we still need to show that, for this choice of ϕ_{of}, P_S is feasible. We get this from feasibility of $\mathcal{B}$-strong fairness of actions, as proven in Proposition 35. We therefore conclude that Theorem 56 applies to Formula 3 and hence we conclude that Theorem 28 is true. ◀

D.4.2 SHFA

We also prove Theorem 29.

▶ **Theorem 29.** *An LTS satisfies Formula 4 if, and only if, its initial state does not admit a $\mathcal{B}$-progressing path that satisfies strong $\mathcal{B}$-hyperfairness of actions and is $(\rho, \alpha_f, \alpha_e)$-violating.*

Formula 4 is:

$$\neg\langle\rho \cdot \overline{\alpha_f}^\star\rangle(\langle\alpha_e\rangle\mathit{tt} \vee [\overline{\mathcal{B}}]\mathit{ff} \vee \bigvee_{\emptyset \neq F \subseteq \overline{\mathcal{B}}} \nu X.(\bigwedge_{a \in F} \mu W.((\bigwedge_{b \in \overline{\mathcal{B}}\setminus F} [\overline{\mathcal{B}}^\star \cdot b]\mathit{ff}) \wedge (\langle a \setminus \alpha_f\rangle X \vee \langle\overline{\alpha_f}\rangle W))))$$

Proof. To get Formula 4 from Formula 8, we take $\phi_{of}(b) = [\overline{\mathcal{B}}^\star \cdot b]\mathit{ff}$ for all $b \in \overline{\mathcal{B}}$. This is a mapping to closed modal μ-calculus formulae. We also need to show that in every $\mathcal{B}$-locked state, $\phi_{of}(b)$ is satisfied for all $b \in \overline{\mathcal{B}}$, and vice versa. In a $\mathcal{B}$-locked state, all enabled actions are blocking. Hence, in order to reach a state where an non-blocking action is enabled, a blocking action must first be taken. Therefore, in a $\mathcal{B}$-locked state all non-blocking actions b are not $\mathcal{B}$-reachable, and so $[\overline{\mathcal{B}}^\star \cdot b]\mathit{ff}$ is satisfied. The other way around, if $[\overline{\mathcal{B}}^\star \cdot b]\mathit{ff}$ is satisfied for all non-blocking actions b, then no non-blocking actions can be enabled since otherwise they would be $\mathcal{B}$-reachable. Hence, all enabled actions must be blocking and it must be a $\mathcal{B}$-locked state.

We show a path satisfies $\mathcal{B}$-strong hyperfairness of actions (Definition 16) exactly when it satisfies P_S interpreted on this ϕ_{of}. We can use that the path is $\mathcal{B}$-progressing, since both the theorem on $\mathcal{B}$-SHFA and the theorem of P_S include this. A path satisfies $\mathcal{B}$-SHFA if,

and only if, for every suffix π' of π, every action $a \in \overline{\mathcal{B}}$ that is relentlessly $\mathcal{B}$-reachable in π' occurs in π'.

Using a similar argument as given in the proof of Theorem 28, we claim that if an action is relentlessly $\mathcal{B}$-reachable on a suffix of a path it is also relentlessly $\mathcal{B}$-reachable on that whole path: if every suffix of a path π has a state where the action a is $\mathcal{B}$-reachable, then so does every suffix π'' of π', since π'' is also a suffix of π. The other way around, if every suffix π'' of π', which is a suffix of π, contains a state where the action a is $\mathcal{B}$-reachable, then since every suffix of π either contains π' or is a suffix of π', every suffix of π also contains such a state. It is also the case for $\mathcal{B}$-SHFA that if a path is $\mathcal{B}$-progressing and finite, no non-blocking action can be relentlessly $\mathcal{B}$-reachable on that path, since in a $\mathcal{B}$-locked state no non-blocking action is $\mathcal{B}$-reachable. Additionally, on infinite paths every action that occurs in every suffix of the path must occur infinitely often.

Using these insights, we can rewrite $\mathcal{B}$-SHFA to the following: a path π satisfies $\mathcal{B}$-SHFA if, and only if, every action $a \in \overline{\mathcal{B}}$ that is relentlessly $\mathcal{B}$-reachable in π occurs infinitely often in π.

If an action is not relentlessly $\mathcal{B}$-reachable on a path, then the path must contain a suffix in which the action is perpetually non-$\mathcal{B}$-reachable, and vice-versa. Hence, $\mathcal{B}$-SHFA requires that for every non-blocking action a, the action either occurs infinitely often on a path or the path has a suffix in which the action is perpetually non-$\mathcal{B}$-reachable. Similar to the $\mathcal{B}$-SFA case, the only difference between this definition and P_S interpreted on $\phi_{of}(b) = [\overline{\mathcal{B}}^\star \cdot b]\mathit{ff}$ for all $b \in \overline{\mathcal{B}}$, is that the $\mathcal{B}$-SHFA definition uses an exclusive or and the P_S definition does not. However, for this choice of ϕ_{of}, this does not matter. If a non-blocking action occurs infinitely often it must be infinitely often enabled, and hence relentlessly $\mathcal{B}$-reachable on the path, thus there is no suffix of the path where the action in perpetually unreachable. In the other direction, if a path has a suffix on which a non-blocking action is perpetually not $\mathcal{B}$-reachable, then the action is only finitely often $\mathcal{B}$-reachable on the path and therefore also only finitely often enabled. Then it can also only occur finitely often.

The final step to proving Theorem 29 is to establish that $\mathcal{B}$-SHFA is feasible. This was proven in Proposition 38. Hence, we can apply Theorem 56 to Formula 4 to conclude Theorem 29 is true.   ◀