Chapter 15

⋆ Temporal Properties

When I eventually met Mr Right I had no idea that his first name was Always.

- Rita Rudner.

The modal logic **HML** of Chapter 13, while faithfully characterising properties which are relevant for distinguishing between process behaviours, has a fundamental drawback: a given formula $P \in \textbf{HML}$ can only explore the initial behaviour of a process, namely its first k steps where $k = md\,(P)$ is the modal depth of the formula. We cannot write a single formula that will explore a process to an unbounded length of its execution.

Consider, for example, the property of being *deadlockable*. In various examples of real-world system verifications, a common question is whether or not the system in question might at some point in time deadlock, that is, reach a state from which no action is possible. For example, we might like to verify that a new operating system design can never get in to a deadlocked state, one in which the machine on which it is running simply "hangs" leaving the user to apply the age-old solution of turning it off and on again.

Such properties are referred to as ***temporal properties***, as they refer to the long-term behaviour of a system throughout the lifetime of its execution. Note that, since these properties are still based on the behaviour of systems, any such property which is true of a given process will be true of any equivalent process. These properties typically fall under one of the following two categories:

- ***Safety properties*** assert that *nothing bad ever happens*. Typical examples of safety properties include: the operating system will never deadlock; or a car will never be able to enter a level crossing at the same time as a train.
- ***Liveness properties*** assert that *something good eventually happens*. Typical examples of liveness properties include: having pressed the elevator button the elevator will eventually arrive; or if a train arrives its signal will eventually turn green.

In this chapter we will explore various standard temporal properties, as well as the means to define our own temporal properties from recursive equations involving operators of the modal logic.

15.1 Three Standard Temporal Operators

A variety of fundamental temporal operators have been devised for expressing properties. Some of these are described as follows.

15.1.1 Always: $\Box P$

The most basic safety property, that *nothing bad ever happens*, is catered for by the temporal operator $\Box P$ (pronounced "box P") which asserts that the property P is true in *every* state into which the process may evolve. Formally,

$$\begin{aligned} E \models \Box P \quad & \text{if, and only if,} \\ & F \models P \text{ for all } F \text{ such that } E \xrightarrow{w} F \text{ for some } w \in A^*. \end{aligned}$$

This is similar to the action "box" operator $[a]P$ except that the transitions involve arbitrary strings $w \in A^*$ rather than a single action $a \in A$.

We can view this property as an infinite conjunction; the property asserts that P is true after *any* number of transitions:

$$P \quad \wedge \quad [-]P \quad \wedge \quad [-][-]P \quad \wedge \quad [-][-][-]P \quad \wedge \quad \cdots$$

That is to say, P is true at the start; and after any single transition; and after any two transitions; and after any three transitions; and .... Another way to view this is as a recursive property: the above infinite conjunction expresses a property X which satisfies the recursive equation

$$X \;=\; P \;\wedge\; [-]X$$

which describes a property expressing the fact that P is true, and no matter what transition happens the property defined by X must hold (that is, P is true, and no matter what transition happens the property defined by X must hold (that is, ...)). Note that *every* one of an infinite number of properties must be true in order to satisfy $\Box P$.

Deadlock-freedom is an example of a property which can be defined with this operator: being free of deadlocks means that the property of being able to perform *some* action, *ie* $\langle - \rangle$true, is true in *every* state into which the process may evolve:

$$\text{Deadlock-free} \;=\; \Box\langle - \rangle\text{true}.$$

Example 15.1

Consider the two clock processes `Clock` and $\texttt{Clock}_\star$ from Example 11.11 pictured in Figure 11.7 (page 298). Recall that these two processes could not be distinguished by any formula of **HML**, since they were n-game equivalent for every $n \in \mathbb{N}$, despite the fact that they are not bisimilar (that is, they are not ∞-game equivalent). What distinguishes $\texttt{Clock}_\star$ from `Clock` is the possibility that it may evolve into a deadlock-free state:

$$\texttt{Clock}_\star \models \langle tick\rangle\Box\langle tick\rangle\mathsf{true} \quad\text{but}\quad \texttt{Clock} \not\models \langle tick\rangle\Box\langle tick\rangle\mathsf{true}.$$

Example 15.2

The safety property for our railway level crossing example of Section 14.3 is that at no time can a car cross at the same time as a train. This is expressed as

$$\Box\Big([\texttt{ccross}]\mathsf{false} \ \vee\ [\texttt{tcross}]\mathsf{false}\Big).$$

That is to say, it is always the case that either I cannot do a `ccross` action (a car cannot cross) or I cannot do a `tcross` action (a train cannot cross).

15.1.2 Possibly: $\Diamond P$

If we wish to express the *possibility* that *something bad may happen*, we can use another standard temporal operator, $\Diamond P$ (pronounced "diamond P"), which asserts that the property P is true in *some* state into which the process may evolve. Formally,

$$E \models \Diamond P \quad \text{if, and only if,}$$
$$F \models P \text{ for some } F \text{ such that } E \xrightarrow{w} F \text{ for some } w \in A^*.$$

This is similar to the action "diamond" operator $\langle a\rangle P$ except that the transitions involve arbitrary strings $w \in A^*$ rather than a single action $a \in A$.

We can view this property as an infinite disjunction; the property asserts that P is true after *some* number of transitions:

$$P \ \vee\ \langle -\rangle P \ \vee\ \langle -\rangle\langle -\rangle P \ \vee\ \langle -\rangle\langle -\rangle\langle -\rangle P \ \vee\ \cdots$$

That is to say, P is true: either at the start; or after some single transition; or after some two transitions; or after some three transitions; or .... Another way to view this is as a recursive property: the above infinite disjunction expresses a property X which satisfies the recursive equation

$$X \ =\ P \ \vee\ \langle -\rangle X$$

which describes a property expressing the fact that either P is true, or there is some transition which may happen after which the property defined by X will hold (that is, either P is true, or there is some transition which may happen after which the property defined by X will hold (that is, $\ldots$)). Note that *some* one of an infinite number of properties must be true in order to satisfy this $\Diamond P$.

The property of being deadlockable, *ie* the opposite of Deadlock-freedom, is an example of a property which can be defined with this operator: being deadlockable means that the property of *not* being able to perform *any* action, *ie* $[-]\mathsf{false}$, is true in *some* state into which the process may evolve:

$$\text{Deadlockable} \;=\; \Diamond[-]\mathsf{false}.$$

The properties $\Diamond P$ and $\Box P$ are related in the same way that $\langle a\rangle P$ and $[a]P$ are related, in that each is used to express the negation of the other:

$$\neg\Diamond P \;=\; \Box\neg P \qquad \text{and} \qquad \neg\Box P \;=\; \Diamond\neg P$$

These operations are thus inter-definable:

- $\Diamond P \;=\; \neg\Box\neg P$: P is true in *some* reachable state if, and only if, it is *not* true that P is *false* in *every* reachable state;
- $\Box P \;=\; \neg\Diamond\neg P$: P is true in *every* reachable state if, and only if, it is *not* true that P is *false* in *some* reachable state.

Exercise 15.2 (Solution on page 488)

Use the above relationships between $\Box$ and $\Diamond$ to show that

$$\text{Deadlock-free} \;=\; \neg\text{Deadlockable}$$

where Deadlock-free $= \Box\langle-\rangle\mathsf{true}$ and Deadlockable $= \Diamond[-]\mathsf{false}$.

15.1.3 Until: $P\,\mathrm{U}\,Q$

It is often desirable to express that some property remains true until some other property becomes true, and that this latter property eventually does at some time become true. For example, we might wish to assert that when we send a document to a printer, the document will remain on the printer queue until it is scheduled to be printed, and it will eventually be printed. This type of property is expressed by the temporal operator $P\,\mathrm{U}\,Q$ which asserts two things: that a particular property Q will eventually be true; and that until that time the property P will remain true. Formally:

$$\begin{array}{ll} E \models P\,\mathrm{U}\,Q & \text{if, and only if,} \\ & \text{if } \; E = E_0 \xrightarrow{a_1} E_1 \xrightarrow{a_2} E_2 \xrightarrow{a_3} \cdots \xrightarrow{a_n} E_n \overset{a}{\not\rightarrow} \\ & \text{or } \; E = E_0 \xrightarrow{a_1} E_1 \xrightarrow{a_2} E_2 \xrightarrow{a_3} E_3 \xrightarrow{a_4} \cdots \\ & \text{then } \; \exists k \text{ such that } E_k \models Q \text{ and } E_i \models P \text{ for all } i < k. \end{array}$$

Note that $P \, \mathrm{U} \, Q$ is true if Q initially holds; and that P can remain true when Q eventually holds but doesn't have to.

We can view the property $P \, \mathrm{U} \, Q$ as a property X which satisfies the recursive equation

$$X \;=\; Q \;\vee\; (P \;\wedge\; \langle - \rangle \mathsf{true} \;\wedge\; [-]X)$$

which describes a property expressing the fact that: either Q is true; or P is true, and it is possible to do something, and no matter what you do the property defined by X must hold (that is: either Q is true; or P is true, and it is possible to do something, and no matter what you do the property defined by X must hold (that is: ...)). Again note that *some* one of an infinite number of properties must be true in order to satisfy $P \, \mathrm{U} \, Q$.

Exercise 15.3 (Solution on page 488)

The generic liveness property asserts that *something good eventually happens.* Show how to express the temporal operator $\mathrm{Ev}\, P$ (pronounced "eventually P") using the above standard temporal operators.

15.2 Recursive Properties

The temporal operators considered in the previous section could all be viewed as solutions to recursive equations over the language **HML** of modal logic. For example, we noted above that $\Box P$ expresses a property X which satisfies the recursive equation

$$X \;=\; P \;\wedge\; [-]X.$$

Thus we would want $E \models \Box P$ to hold if, and only if, the following is true:

$$E \models X \;\text{ if, and only if, }\; E \models P \wedge [-]X.$$

To incorporate this idea into the logic **HML**, we need to introduce variables such as X into the language of properties. However, the semantic definition from Section 13.2 gives us no means by which we can determine if $E \models X$. It is not enough to assume that each variable X is defined by some equation $X = P$ and declare that $E \models X$ if, and only if, $E \models P$. For example, if $E \stackrel{\text{def}}{=} a.E$ and X is defined by $X = \langle a \rangle X$, we would only be able to infer that $E \models X$ if, and only if, $E \models X$; either answer – $E \models X$ or $E \not\models X$ – is consistent with this observation.

To get around this deficiency, we need to introduce some mechanism to determine which states satisfy a variable property like X. This is provided by a ***valuation function***

$$\mathsf{V} : \text{Variables} \to \mathcal{P}\,(\text{States})$$

where Variables is a set of variables (such as X above), and States is the set of states of the labelled transition system in which we are interested. Modal formulæ involving variables are then interpreted with respect to a valuation function as follows:

$E \models_{\mathsf{V}} \mathsf{true}$	for *all* E.
$E \models_{\mathsf{V}} \mathsf{false}$	for *no* E.
$E \models_{\mathsf{V}} X$	if, and only if, $E \in \mathsf{V}(X)$.
$E \models_{\mathsf{V}} \neg P$	if, and only if, $E \not\models_{\mathsf{V}} P$.
$E \models_{\mathsf{V}} P \wedge Q$	if, and only if, $E \models_{\mathsf{V}} P$ *and* $E \models_{\mathsf{V}} Q$.
$E \models_{\mathsf{V}} P \vee Q$	if, and only if, $E \models_{\mathsf{V}} P$ *or* $E \models_{\mathsf{V}} Q$.
$E \models_{\mathsf{V}} \langle a \rangle P$	if, and only if, $F \models_{\mathsf{V}} P$ for *some* F such that $E \xrightarrow{a} F$.
$E \models_{\mathsf{V}} [a] P$	if, and only if, $F \models_{\mathsf{V}} P$ for *all* F such that $E \xrightarrow{a} F$.

This is identical to the original definition for $E \models P$ but for the extra clause which determines when a state E satisfies a variable property X; this case is catered for by the valuation function V which is now attached to the satisfaction relation $\models_{\mathsf{V}}$.

In a similar fashion we can extend the global semantic definition $\|P\|$ to incorporate the valuation function as follows.

$$\begin{aligned}
\|\mathsf{true}\|_{\mathsf{V}} &= \text{States} \\
\|\mathsf{false}\|_{\mathsf{V}} &= \emptyset \\
\|X\|_{\mathsf{V}} &= \mathsf{V}(X) \\
\|\neg P\|_{\mathsf{V}} &= \overline{\|P\|_{\mathsf{V}}} \\
\|P \wedge Q\|_{\mathsf{V}} &= \|P\|_{\mathsf{V}} \cap \|Q\|_{\mathsf{V}} \\
\|P \vee Q\|_{\mathsf{V}} &= \|P\|_{\mathsf{V}} \cup \|Q\|_{\mathsf{V}} \\
\|\langle a \rangle P\|_{\mathsf{V}} &= \{\, E \in \text{States} \;:\; E \xrightarrow{a} E' \text{ for some } E' \in \|P\|_{\mathsf{V}} \,\} \\
\|[a] P\|_{\mathsf{V}} &= \{\, E \in \text{States} \;:\; E \xrightarrow{a} E' \text{ implies } E' \in \|P\|_{\mathsf{V}} \,\}
\end{aligned}$$

Theorem 13.12 then extends directly to properties with variables as follows.

Theorem 15.3

$E \models_{\mathsf{V}} P$ *if, and only if,* $E \in \|P\|_{\mathsf{V}}$.

Exercise 15.4 (Solution on page 489)

Prove Theorem 15.3

15.2.1 Solving Recursive Equations

In order to determine if a state E satisfies the property being expressed by a recursive equation $X = P$, where P is an **HML** formula possibly involving the variable X, we need to somehow "solve" the equation $X = P$. This equation simply declares that the set of states which satisfy the property X being defined is precisely the set of states which satisfy the property P; in other words, we need to equate the following two sets:

$$||X||_{\mathsf{V}} = ||P||_{\mathsf{V}}.$$

To solve this equation we need to find a valuation V which makes this a valid set equation. Since $||X||_{\mathsf{V}} = \mathsf{V}(X)$, the answer we seek is the set S which such a valuation V assigns to the variable X. That is, the solution is a set $S \subseteq \mathsf{States}$ satisfying

$$S \;=\; ||P||_{\mathsf{V}[X \mapsto S]}$$

where $\mathsf{V}[X \mapsto S]$ denotes the valuation V updated by assigning the set S to the variable X:

$$\big(\mathsf{V}[X \mapsto S]\big)(Y) \;=\; \begin{cases} S & \text{if } Y = X \\ \mathsf{V}(Y) & \text{if } Y \neq X. \end{cases}$$

Example 15.4

Consider a property X which satisfies the equation

$$X = \langle a \rangle X.$$

Informally, this equation suggests that an infinite sequence of consecutive a actions can be performed:

$$\begin{aligned} E \models X \;&\Leftrightarrow\; E \xrightarrow{a} E' \text{ for some } E' \text{ such that } E' \models X \\ &\Leftrightarrow\; E \xrightarrow{a} E' \xrightarrow{a} E'' \text{ for some } E' \text{ and } E'' \text{ such that } E'' \models X \\ &\Leftrightarrow\; \cdots \\ &\Leftrightarrow\; E \xrightarrow{a} E' \xrightarrow{a} E'' \xrightarrow{a} E''' \xrightarrow{a} \cdots \text{ for some } E', E'', E''', \ldots \end{aligned}$$

Let $S \subseteq \mathsf{States}$ be the set of such states:

$$S = \{\, E \in \mathsf{States} \;:\; E \xrightarrow{a} \cdot \xrightarrow{a} \cdot \xrightarrow{a} \cdots \,\}.$$

Then

$$||\langle a \rangle X||_{\mathsf{V}[X \mapsto S]} \;=\; \{\, E \in \mathsf{States} \;:\; E \xrightarrow{a} E' \text{ for some } E' \in S \,\} \;=\; S.$$

Thus, as intended, S satisfies the equation $S \;=\; ||\langle a \rangle X||_{\mathsf{V}[X \mapsto S]}$.

One problem that we have is that an arbitrary recursive equation needn't necessarily have a solution. For example, if we take the recursive equation

$$X = \neg X$$

then given any valuation V,

$$\begin{aligned} \|X\|_{\mathsf{V}} &= \mathsf{V}(X) \\ &\neq \overline{\mathsf{V}(X)} = \overline{\|X\|_{\mathsf{V}}} = \|\neg X\|_{\mathsf{V}}. \end{aligned}$$

Another problem is that an equation may be satisfied by many different solutions, as illustrated in the following.

Exercise 15.5 (Solution on page 489)

Show that the set $S = \emptyset$ satisfies the equation $S = \|\langle a\rangle X\|_{\mathsf{V}[X\mapsto S]}$ from Example 15.4.

However, we will show here that any recursive equations which does not involve negation has a solution, and moreover we will show how to solve it to obtain the intended solution.

15.2.2 Fixed Point Solutions

Let $f : \mathcal{P}\,(\mathsf{States}) \to \mathcal{P}\,(\mathsf{States})$ be defined by

$$f(S) = \|P\|_{\mathsf{V}[X\mapsto S]}.$$

Then a solution to the recursive equation $X = P$ is merely a ***fixed point*** of this function: a set $S \subseteq \mathsf{States}$ such that $S = f(S)$. By the Knaster-Tarski Theorem (Theorem 6.18, page 174), this function is guaranteed to have a fixed point – in fact both greatest and least fixed points – so long as the function is monotonic. That this function is monotonic is an immediate corollary of the following result.

Theorem 15.5

Let P be an **HML** *formula which does not involve negation, and let* V *and* W *be valuations such that* $\mathsf{V}(X) \subseteq \mathsf{W}(X)$ *for all* X*. Then* $\|P\|_{\mathsf{V}} \subseteq \|P\|_{\mathsf{W}}$.

Proof: By induction – and arguing by cases – on the structure of P.

$\underline{P = \textbf{true}}$: $\|\mathsf{true}\|_{\mathsf{V}} = \mathsf{States} = \|\mathsf{true}\|_{\mathsf{W}}$.

$\underline{P = \textbf{false}}$: $\|\mathsf{false}\|_{\mathsf{V}} = \emptyset = \|\mathsf{false}\|_{\mathsf{W}}$.

$\underline{P = X}$: $\|X\|_{\mathsf{V}} = \mathsf{V}(X) \subseteq \mathsf{W}(X) = \|X\|_{\mathsf{W}}$.

$\underline{P = Q_1 \wedge Q_2}$:
$$\begin{aligned} \|Q_1 \wedge Q_2\|_{\mathsf{V}} &= \|Q_1\|_{\mathsf{V}} \cap \|Q_2\|_{\mathsf{V}} \\ &\subseteq \|Q_1\|_{\mathsf{W}} \cap \|Q_2\|_{\mathsf{W}} \\ &= \|Q_1 \wedge Q_2\|_{\mathsf{W}} \end{aligned}$$

$$\begin{aligned}\underline{P = Q_1 \vee Q_2}: \quad \|Q_1 \vee Q_2\|_{\mathsf{V}} &= \|Q_1\|_{\mathsf{V}} \cup \|Q_2\|_{\mathsf{V}} \\ &\subseteq \|Q_1\|_{\mathsf{W}} \cup \|Q_2\|_{\mathsf{W}} \\ &= \|Q_1 \vee Q_2\|_{\mathsf{W}}\end{aligned}$$

$$\begin{aligned}\underline{P = \langle a\rangle Q}: \quad \|\langle a\rangle Q\|_{\mathsf{V}} &= \{\, E \in \mathsf{States} \ : \ E \xrightarrow{a} E' \text{ such that } E' \in \|Q\|_{\mathsf{V}} \} \\ &\subseteq \{\, E \in \mathsf{States} \ : \ E \xrightarrow{a} E' \text{ such that } E' \in \|Q\|_{\mathsf{W}} \} \\ &= \|\langle a\rangle Q\|_{\mathsf{W}}\end{aligned}$$

$$\begin{aligned}\underline{P = [a]Q}: \quad \|[a]Q\|_{\mathsf{V}} &= \{\, E \in \mathsf{States} \ : \ E \xrightarrow{a} E' \text{ implies } E' \in \|Q\|_{\mathsf{V}} \} \\ &\subseteq \{\, E \in \mathsf{States} \ : \ E \xrightarrow{a} E' \text{ implies } E' \in \|Q\|_{\mathsf{W}} \} \\ &= \|[a]Q\|_{\mathsf{W}} \qquad \square\end{aligned}$$

The Knaster-Tarski Theorem thus tells us that recursive equations which do not involve negation have two identifiable solutions, corresponding to their greatest and least fixed point solutions. This begs the question, when considering a recursively-defined property, as to *which* solution – if indeed either of them – represents the *intended* solution. That is, if we express a property as a recursive equation $X = P$, the set of states which satisfy the property we have in mind is a fixed point of the function $f(S) = \|P\|_{\mathsf{V}[X\mapsto S]}$; but is it the greatest fixed point, or the least fixed point, or some fixed point in between?

This question will be explored in Section 15.4, where we will show that the answer is roughly:

- least fixed points express liveness properties; and
- greatest fixed points express safety properties.

Before we do this, though, we first look more carefully at adding these two fixed points to the modal logic **HML** without negation. The resulting logic with fixed points is called the ***modal mu-calculus***, and is one of the most fundamental logics used in the specification of computer systems.

Exercise 15.6 (Solution on page 490)

What are the least and greatest fixed points of the function

$$f(S) = \|\langle a\rangle X\|_{\mathsf{V}[X\mapsto S]}$$

corresponding to the property considered in Example 15.4?

Can you find a fixed point which is neither least nor greatest?

$$
\begin{aligned}
||\mathsf{true}||_{\mathsf{V}} &= \mathsf{States} \\
||\mathsf{false}||_{\mathsf{V}} &= \emptyset \\
||P \wedge Q||_{\mathsf{V}} &= ||P||_{\mathsf{V}} \cap ||Q||_{\mathsf{V}} \\
||P \vee Q||_{\mathsf{V}} &= ||P||_{\mathsf{V}} \cup ||Q||_{\mathsf{V}} \\
||\langle a\rangle P||_{\mathsf{V}} &= \{\, E \in \mathsf{States} \;:\; E \xrightarrow{a} E' \text{ for some } E' \in ||P||_{\mathsf{V}} \,\} \\
||[a]P||_{\mathsf{V}} &= \{\, E \in \mathsf{States} \;:\; E \xrightarrow{a} E' \text{ implies } E' \in ||P||_{\mathsf{V}} \,\} \\
||X||_{\mathsf{V}} &= \mathsf{V}(X) \\
||\mu X.P||_{\mathsf{V}} &= \bigcap \Big\{\, S \subseteq \mathsf{States} \;:\; S \supseteq ||P||_{\mathsf{V}[X\mapsto S]} \,\Big\} \\
||\nu X.P||_{\mathsf{V}} &= \bigcup \Big\{\, S \subseteq \mathsf{States} \;:\; S \subseteq ||P||_{\mathsf{V}[X\mapsto S]} \,\Big\}
\end{aligned}
$$

Figure 15.1: Global semantics of the modal mu-calculus.

15.3 The Modal Mu-Calculus

The syntax of the modal mu-calculus consists of the logic **HML** – minus negation – extended with variables and constructs for defining both greatest and least fixed points. Formally, it is presented by the following BNF equation:

$$
\begin{aligned}
P, Q \;::=\; & \mathsf{true} \;\mid\; \mathsf{false} \;\mid\; P \wedge Q \;\mid\; P \vee Q \;\mid\; \langle a\rangle P \;\mid\; [a]P \\
\mid\; & X \;\mid\; \mu X.P \;\mid\; \nu X.P
\end{aligned}
$$

The symbols μ and ν are the characters "mu" and "nu" from the Greek alphabet (from which the name "mu-calculus" derives). The formula $\mu X.P$ is used to represent the least fixed point of the equation $X = P$ (or more correctly, of the function $f(S) = ||P||_{\mathsf{V}[X\mapsto S]}$) whereas $\nu X.P$ is used to represent its greatest fixed point.

An inductive definition of the semantic function $||P||_{\mathsf{V}}$ – defining which states satisfy the property P with respect to the valuation V – is given in Figure 15.1. The clauses are identical to those presented for the basic modal logic **HML** in Figure 13.2 with the inclusion of the obvious clause for variables, and clauses for the fixed points as given by the Knaster-Tarski Theorem (Theorem 6.18, page 174), That the Knaster-Tarski Theorem applies follows from the fact that the function $f(S) = ||P||_{\mathsf{V}[X\mapsto S]}$ is monotonic. We demonstrated this for **HML** in Theorem 15.5, but we need to extend this result for the larger logic.

Theorem 15.6

Let P be a modal mu-calculus formula which does not involve negation, and let V *and* W *be valuations such that* $\mathsf{V}(X) \subseteq \mathsf{W}(X)$ *for all* X. *Then* $\|P\|_{\mathsf{V}} \subseteq \|P\|_{\mathsf{W}}$.

Proof: By induction – and arguing by cases – on the structure of P. All of the cases have been catered for in the proof of Theorem 15.5 – and carry over directly to the present setting – apart from the cases of variables and fixed point formulæ, which we handle here.

$\underline{P = X}$: $\|X\|_{\mathsf{V}} \;=\; V(X) \;\subseteq\; W(X) \;=\; \|X\|_{\mathsf{W}}$

$\underline{P = \mu X.Q}$:
$$\begin{aligned} E \in \|\mu X.Q\|_{\mathsf{V}} &\Leftrightarrow E \in S \text{ whenever } \|Q\|_{\mathsf{V}[X\mapsto S]} \subseteq S \\ &\Rightarrow E \in S \text{ whenever } \|Q\|_{\mathsf{W}[X\mapsto S]} \subseteq S \\ &\Leftrightarrow E \in \|\mu X.Q\|_{\mathsf{W}} \end{aligned}$$

$\underline{P = \nu X.Q}$:
$$\begin{aligned} E \in \|\nu X.Q\|_{\mathsf{V}} &\Leftrightarrow E \in S \text{ for some } S \text{ where } S \subseteq \|Q\|_{\mathsf{V}[X\mapsto S]} \\ &\Rightarrow E \in S \text{ for some } S \text{ where } S \subseteq \|Q\|_{\mathsf{W}[X\mapsto S]} \\ &\Leftrightarrow E \in \|\nu X.Q\|_{\mathsf{W}} \end{aligned}$$
□

Definition 15.6

A direct definition of when a state E satisfies a property P of the modal mu-calculus with respect to a valuation V *for interpreting free variables which appear in P is as follows:*

$E \models_{\mathsf{V}}$ *true*	*for* all E.
$E \models_{\mathsf{V}}$ *false*	*for* no E.
$E \models_{\mathsf{V}} P \wedge Q$	*if, and only if,* $E \models_{\mathsf{V}} P$ and $E \models_{\mathsf{V}} Q$.
$E \models_{\mathsf{V}} P \vee Q$	*if, and only if,* $E \models_{\mathsf{V}} P$ or $E \models_{\mathsf{V}} Q$.
$E \models_{\mathsf{V}} \langle a \rangle P$	*if, and only if,* $F \models_{\mathsf{V}} P$ *for* some F *such that* $E \xrightarrow{a} F$.
$E \models_{\mathsf{V}} [a] P$	*if, and only if,* $F \models_{\mathsf{V}} P$ *for* all F *such that* $E \xrightarrow{a} F$.
$E \models_{\mathsf{V}} X$	*if, and only if,* $E \in \mathsf{V}(X)$.
$E \models_{\mathsf{V}} \mu X.P$	*if, and only if,* $\forall S \subseteq \text{States}:$ *if* $E \notin S$ *then* $\exists F \notin S$ *such that* $F \models_{\mathsf{V}[X\mapsto S]} P$
$E \models_{\mathsf{V}} \nu X.P$	*if, and only if,* $\exists S \subseteq \text{States}:$ $E \in S$ *and* $\forall F \in S : F \models_{\mathsf{V}[X\mapsto S]} P$

We leave it as an exercise (Exercise 5, page 402) to prove (by induction on the structure of P) that $E \models_V P$ if, and only if, $E \in \|P\|_V$.

Leaving negation out of the logic is not a real restriction, as the result from Section 13.3 that negation is definable in the modal logic **HML** extends to the whole of the modal mu-calculus. This is justified by the following.

Exercise 15.7 (Solution on page 490)

The negation of a modal mu-calculus formula can be inductively defined as follows:

$$\begin{aligned} \mathsf{neg}(\mathsf{true}) &= \mathsf{false} & \mathsf{neg}(\langle a\rangle P) &= [a]\mathsf{neg}(P) \\ \mathsf{neg}(\mathsf{false}) &= \mathsf{true} & \mathsf{neg}([a]P) &= \langle a\rangle\mathsf{neg}(P) \\ \mathsf{neg}(P \wedge Q) &= \mathsf{neg}(P) \vee \mathsf{neg}(Q) & \mathsf{neg}(\mu X.P) &= \nu X.\mathsf{neg}(P) \\ \mathsf{neg}(P \vee Q) &= \mathsf{neg}(P) \wedge \mathsf{neg}(Q) & \mathsf{neg}(\nu X.P) &= \mu X.\mathsf{neg}(P) \\ \mathsf{neg}(X) &= X \end{aligned}$$

Prove that $E \models_{\overline{V}} \mathsf{neg}(P)$ if, and only if, $E \not\models_V P$, where $\overline{V}(X) = \overline{V(X)}$.

15.4 Least versus Greatest Fixed Points

We now understand how to interpret recursive logical properties as fixed points of particular functions between sets of states. However, we are left with the problem of understanding why the property we intend is expressed by either the greatest or the least fixed point of this function, as well as the problem of knowing which. To solve this, we shall explore how such a recursive property can be understood by "unrolling" it.

Given a property X defined by a recursive equation $X = P$, we can unroll the equation by replacing each occurrence of X on the right-hand-side by P itself. Clearly this will not change the meaning of the property being defined by the recursive equation.

Example 15.7

Suppose we wish to express the property that an infinite sequence of consecutive a actions can happen. That is, denoting this property by X, we would like the following to be the case:

$$E \models X \text{ if, and only if, } E \xrightarrow{a} \cdot \xrightarrow{a} \cdot \xrightarrow{a} \cdots .$$

This property is expressed by the recursive equation

$$X \;=\; \langle a\rangle X$$

which we can repeatedly unroll as follows:

$$\begin{aligned} X &= \langle a\rangle X \\ &= \langle a\rangle\langle a\rangle X \\ &= \langle a\rangle\langle a\rangle\langle a\rangle X \\ &= \langle a\rangle\langle a\rangle\langle a\rangle\langle a\rangle X \\ &= \langle a\rangle\langle a\rangle\langle a\rangle\langle a\rangle\langle a\rangle \cdots \end{aligned}$$

Example 15.8

Suppose we wish to express the property that an a action must eventually occur. This property is expressed by the recursive equation

$$X \;=\; \langle -\rangle\mathsf{true} \;\wedge\; [-a]X.$$

That is: some action is possible; and if anything other than an a action occurs, then an a action must eventually occur in the resulting state. We can repeatedly unroll this recursive equation as follows:

$$\begin{aligned} X &= \langle -\rangle\mathsf{true} \;\wedge\; [-a]X \\ &= \langle -\rangle\mathsf{true} \;\wedge\; [-a]\Big(\langle -\rangle\mathsf{true} \;\wedge\; [-a]X\Big) \\ &= \langle -\rangle\mathsf{true} \;\wedge\; [-a]\Big(\langle -\rangle\mathsf{true} \;\wedge\; [-a](\langle -\rangle\mathsf{true} \;\wedge\; [-a]X)\Big) \\ &= \langle -\rangle\mathsf{true} \;\wedge\; [-a]\Big(\langle -\rangle\mathsf{true} \;\wedge\; [-a](\langle -\rangle\mathsf{true} \;\wedge\; [-a](\cdots))\Big) \end{aligned}$$

15.4.1 Approximating Fixed Points

By repeatedly unrolling a recursive equation, we seem to eliminate the variable from the formula. Of course we would have to unroll the equation infinitely often in order to get rid of the variable altogether. However, we don't have any means for determining whether or not an infinitely-long property is satisfied. We can, however, define better and better approximations for such properties, by replacing the variable in the rolled-out formula by either false or true. To this end, we can define the nth mu- and nu-approximants as follows.

Definition 15.8

*Given a recursive equation $X = P$, the nth **mu-approximant** $\mu^n X.P$ and the nth **nu-approximant** $\nu^n X.P$ are defined inductively as follows:*

$$\mu^0 X.P = \mathit{false} \qquad \nu^0 X.P = \mathit{true}$$

$$\mu^{n+1} X.P = P[X \mapsto \mu^n X.P] \qquad \nu^{n+1} X.P = P[X \mapsto \nu^n X.P]$$

These definitions extend to all ordinal numbers (see Section 12.5.1), with the following definitions for the approximants corresponding to a limit ordinal λ:

$$\mu^\lambda X.P = \bigvee_{\alpha<\lambda} \mu^\alpha X.P \qquad \nu^\lambda X.P = \bigwedge_{\alpha<\lambda} \nu^\alpha X.P$$

Example 15.9

Recall the property from Example 15.7 that an infinite sequence of consecutive a actions can happen:

$$X = \langle a \rangle X.$$

Its mu-approximants Φ_n and nu-approximants Ψ_n are as follows:

$$\begin{array}{lll} \Phi_0 & = & \mathsf{false} \\ \Phi_1 & = & \langle a \rangle \mathsf{false} \\ \Phi_2 & = & \langle a \rangle \langle a \rangle \mathsf{false} \\ \Phi_3 & = & \langle a \rangle \langle a \rangle \langle a \rangle \mathsf{false} \\ & \vdots & \end{array} \qquad \begin{array}{lll} \Psi_0 & = & \mathsf{true} \\ \Psi_1 & = & \langle a \rangle \mathsf{true} \\ \Psi_2 & = & \langle a \rangle \langle a \rangle \mathsf{true} \\ \Psi_3 & = & \langle a \rangle \langle a \rangle \langle a \rangle \mathsf{true} \\ & \vdots & \end{array}$$

Clearly none of the mu-approximants Φ_n can be satisfied by any state. However, every one of the nu-approximants Ψ_n must be satisfied in order for our intended property to be satisfied.

This is suggestive of a *safety property*: checking that *something bad never happens* (in this case, that an a action is impossible) amounts to checking the validity of *every* unrolling of the formula, starting from true.

Example 15.10

Recall the property from Example 15.7 that an a action must eventually occur:

$$X = \langle - \rangle \mathit{true} \wedge [-a]X.$$

Its mu-approximants Φ_n and nu-approximants Ψ_n are as follows:

$$\begin{aligned}
\Phi_0 &= \mathsf{false}\\
\Phi_1 &= \langle -\rangle\mathsf{true} \ \wedge\ [-a]\mathsf{false}\\
\Phi_2 &= \langle -\rangle\mathsf{true} \ \wedge\ [-a]\big(\langle -\rangle\mathsf{true} \ \wedge\ [-a]\mathsf{false}\big)\\
\Phi_3 &= \langle -\rangle\mathsf{true} \ \wedge\ [-a]\big(\langle -\rangle\mathsf{true} \ \wedge\ [-a](\langle -\rangle\mathsf{true} \ \wedge\ [-a]\mathsf{false})\big)\\
&\vdots
\end{aligned}$$

$$\begin{aligned}
\Psi_0 &= \mathsf{true}\\
\Psi_1 &= \langle -\rangle\mathsf{true} \ \wedge\ [-a]\mathsf{true}\\
\Psi_2 &= \langle -\rangle\mathsf{true} \ \wedge\ [-a]\big(\langle -\rangle\mathsf{true} \ \wedge\ [-a]\mathsf{true}\big)\\
\Psi_3 &= \langle -\rangle\mathsf{true} \ \wedge\ [-a]\big(\langle -\rangle\mathsf{true} \ \wedge\ [-a](\langle -\rangle\mathsf{true} \ \wedge\ [-a]\mathsf{true})\big)\\
&\vdots
\end{aligned}$$

With a little thought, it is apparent that one of the mu-approximants Φ_n must be satisfied in order for our intended property to be satisfied. However, every one of the nu-approximants is satisfied, for example, by a process which runs forever without ever doing an a action.

This is indicative of a *liveness property*: checking that *something good eventually happens* (in this case, that an a action occurs) amounts to checking the validity of *some* unrolling of the formula, starting from false.

In the first of the above two examples, the property which we wished to express was interpreted as the conjunction of all of the nu-approximants (the unrollings starting from true); while in the second of the two examples, the property of interest was interpreted as the disjunction of all of the mu-approximants (the unrollings starting from false). In what follows, we shall see that the first corresponds to the greatest fixed point interpretation of the recursive property, while the second corresponds to the least fixed point interpretation.

In Section 6.5 we described how the least and greatest fixed points of a monotonic function f defined on the powerset of a given set S could be constructed, by repeatedly applying the function f to either the empty set $\emptyset$ (for the least fixed point) or to the whole set S (for the greatest fixed point); this result was given in Theorem 6.19. This is just the result we are looking for here, as the nth mu- and nu-approximants correspond, respectively, to applying the relevant function n times either to the empty set $\emptyset$ or to the whole set States. These facts are recorded in the following.

Theorem 15.10

$f^n(\emptyset) = ||\mu^n X.P||_{\mathsf{V}}$ and $f^n(\mathsf{States}) = ||\nu^n X.P||_{\mathsf{V}}$, where $f(S) = ||P||_{\mathsf{V}[X \mapsto S]}$.

Proof: We prove only the first result, by induction on n, and leave the proof of the second as an exercise (Exercise 6, page 402).

For the base case $n = 0$, we have

$$f^0(\emptyset) \;=\; \emptyset \;=\; ||\mathsf{false}||_{\mathsf{V}} \;=\; ||\mu^0 X.P||_{\mathsf{V}}$$

For the induction step, we have that

$$\begin{aligned} f^{n+1}(\emptyset) &= f(f^n(\emptyset)) \\ &= f(||\mu^n X.P||_{\mathsf{V}}) && \textit{(by induction)} \\ &= ||P||_{\mathsf{V}[X \mapsto ||\mu^n X.P||_{\mathsf{V}}]} \\ &= ||P[X \mapsto \mu^n X.P]||_{\mathsf{V}} \\ &= ||\mu^{n+1} X.P||_{\mathsf{V}} && \Box \end{aligned}$$

Example 15.11

Consider the recursive equation for $\Box P$, the property that P holds in every reachable state:

$$X \;=\; P \;\wedge\; [-]X.$$

This gives rise to the function

$$f(S) \;=\; \{\, E \in ||P||_{\mathsf{V}} \;:\; E \longrightarrow E' \text{ implies } E' \in S \,\}.$$

Using the construction from Theorem 6.19 (page 175) starting from the empty set $\emptyset$, we discover that

$$f(\emptyset) \;=\; \emptyset$$

demonstrating that the least fixed point is the empty set. This certainly does not correspond to the property $\Box P$. However, starting from the universal set States, we get that

$$\begin{aligned} f^0(\mathsf{States}) &= \mathsf{States} \\ f^1(\mathsf{States}) &= ||P||_{\mathsf{V}} \\ f^2(\mathsf{States}) &= \{\, E \in ||P||_{\mathsf{V}} \;:\; E \longrightarrow E' \text{ implies } E' \in ||P||_{\mathsf{V}} \,\} \\ f^3(\mathsf{States}) &= \{\, E \in ||P||_{\mathsf{V}} \;:\; E \longrightarrow E' \text{ or } E \longrightarrow\longrightarrow E' \\ &\qquad\qquad \text{implies } E' \in ||P||_{\mathsf{V}} \,\} \\ &\;\;\vdots \\ f^n(\mathsf{States}) &= \text{states in which } P \text{ is true throughout} \\ &\quad\; \text{the duration of the first } n \text{ transitions.} \end{aligned}$$

This sequence is approaching the set S of sets in which P is true in every reachable state, which is the desired solution to our recursive equation and easily seen to be a fixed point of the function f.

Example 15.12

Consider the recursive equation expressing that a process is deadlockable:

$$X \;=\; [-]\mathsf{false} \;\vee\; \langle - \rangle X$$

This gives rise to the function

$$f(S) \;=\; \{\, E \in \mathsf{States} \;:\; E \not\rightarrow \;\text{ or } E \longrightarrow E' \text{ with } E' \in S \,\}.$$

Using the construction from Theorem 6.19 starting from the universal set States, we discover that

$$f(\mathsf{States}) \;=\; \mathsf{States}$$

demonstrating that the greatest fixed point is the set of all states. This certainly does not correspond to the property that a process is deadlockable. However, starting from the empty set $\emptyset$, we get that

$$\begin{aligned}
f^0(\emptyset) &= \emptyset \\
f^1(\emptyset) &= \{E \in \mathsf{States} \;:\; E \not\rightarrow\} \\
f^2(\emptyset) &= \{E \in \mathsf{States} \;:\; E \not\rightarrow \text{ or } E \longrightarrow E' \not\rightarrow \;\} \\
f^3(\emptyset) &= \{E \in \mathsf{States} \;:\; E \not\rightarrow \text{ or } E \longrightarrow E' \not\rightarrow \\
&\qquad\qquad\qquad\qquad \text{ or } E \longrightarrow E' \longrightarrow E'' \not\rightarrow \;\} \\
&\;\;\vdots \\
f^n(\emptyset) &= \;\text{ states which can deadlock within the first } n \text{ transitions.}
\end{aligned}$$

This sequence is approaching the set S of states that can deadlock, which is the desired solution to our recursive equation and easily seen to be a fixed point of the function f.

15.5 Expressing Standard Temporal Operators

The intuition which you should have drawn from above is the following.

- Greatest fixed point properties are those for which you need to unroll the underlying recursive equation top-down (from true, or the full set of states) an *infinite* number of times in order to verify that the property is always true; if the property fails for some finite unrolling, then the fixed point property itself fails.

In this sense, greatest fixed point properties are representative of ***safety properties*** which assert that *nothing bad ever happens.*

- Least fixed point properties are those for which you need to unroll the underlying recursive equation bottom-up (from false, or the empty set of states) a *finite* number of times in order to verify that the property is eventually true; if the property fails for every finite unrolling, then the fixed point property itself fails.

  In this sense, least fixed point properties are representative of ***liveness properties*** which assert that *something good eventually happens.*

We now consider how to express each of the standard temporal operators introduced in Section 15.1 in the mu-calculus.

15.5.1 Always: $\Box P$

The temporal operator $\Box P$, expressing that the property P is true in every state into which the process may evolve, is defined by the recursive equation

$$X \;=\; P \;\wedge\; [-]X.$$

In order to establish the truth of $\Box P$, the recursive equation would need to be unrolled forever, to make sure nothing goes wrong. As such, this property is expressed by the *greatest* fixed point formula:

$$\Box P \;=\; \nu X.P \;\wedge\; [-]X.$$

15.5.2 Possibly: $\Diamond P$

The temporal operator $\Diamond P$, expressing that the property P is true in some state into which the process may evolve, is defined by the recursive equation

$$X \;=\; P \;\vee\; \langle - \rangle X.$$

In order to establish the truth of $\Diamond P$, the recursive equation would need to be unrolled only until the property can be verified – that is, only until the property P becomes true. As such, this property is expressed by the *least* fixed point formula:

$$\Diamond P \;=\; \mu X.P \;\vee\; \langle - \rangle X.$$

15.5.3 Until: $P \,\mathrm{U}\, Q$

The temporal operator $P \,\mathrm{U}\, Q$, expressing that the property P remains true until the property Q becomes true, which it must eventually do, is defined by the recursive equation

$$X \;=\; Q \;\vee\; (P \;\wedge\; \langle -\rangle \mathsf{true} \;\wedge\; [-]X)$$

In order to establish the truth of $P \,\mathrm{U}\, Q$, the recursive equation would need to be unrolled only until the property can be verified – that is, only until the property Q becomes true (verifying along the way that P remains true). As such, this property is expressed by the *least* fixed point formula:

$$P \,\mathrm{U}\, Q \;=\; \mu X. Q \;\vee\; (P \;\wedge\; \langle -\rangle \mathsf{true} \;\wedge\; [-]X).$$

Exercise 15.12 (Solution on page 491)

1. $\Box P \;=\; \nu Z. P \wedge [-]Z$ means P holds in *every* state.

   What does $\mu Z. P \wedge [-]Z$ mean?
2. $\Diamond P \;=\; \mu Z. P \vee \langle -\rangle Z$ means P holds in *some* (reachable) state.

   What does $\nu Z. P \vee \langle -\rangle Z$ mean?
3. $P \,\mathrm{U}\, Q \;=\; \mu Z. Q \vee \big(P \wedge \langle -\rangle \mathsf{true} \wedge [-]Z\big)$ means Q *will* become true, and until then P will remain true.

   What does $\nu Z. Q \vee \big(P \wedge \langle -\rangle \mathsf{true} \wedge [-]Z\big)$ mean?

15.6 Further Fixed Point Properties

In this section we look at a collection of properties that can be expressed in the mu-calculus.

There is an a^ω path.

By this, we mean that we can do an infinite number of consecutive a transitions starting from the state in question.

If we let X represent this property, then X satisfies the recursive equation

$X \;=\; \langle a\rangle X$ *(It is possible to do an a transition and go to a state in which the property holds.)*

As we are clearly wanting to unroll this fixed point equation infinitely often, to verify that the property holds forever, we are in this case interested in the greatest fixed point solution:

$$\nu X. \langle a\rangle X.$$

There is no a^ω path.

This is the negation of the previous property, and the most straightforward way to find a mu-calculus formula which expresses it is to use the construction from Exercise 15.7:

$$\mathsf{neg}(\nu X.\langle a\rangle X) \;=\; \mu X.[a]X.$$

Unrolling the underlying recursive equation

$X \;=\; [a]X$ *(If I do an a transition, I must end up in a state in which the property holds.)*

suggests an exploration of each a^ω path; as this is a least fixed point property, this search must terminate, namely at a state in which an a transition is not possible.

P holds at every state along some a^ω path.

This is a simple adaptation of the first property above:

$$\nu X.P \;\wedge\; \langle a\rangle X.$$

P holds somewhere along some a^ω path.

We note first that we want to get to a state at which the property P holds by only doing a transitions. This property is expressed by the recursive equation

$X \;=\; P \vee \langle a\rangle X$ *(Either P is true, or it is possible to do an a transition and end up in a state in which the property holds.)*

As we need P to be true at some point, this is a least fixed point property:

$$\mu X.P \;\vee\; \langle a\rangle X.$$

This is not the end of the story, as we require that this path of a transitions leading up to the state in which P is true be the start of an a^ω path. In other words, the point at which P is true must be the start of an a^ω path – which is the first property we considered above: $\nu X.\langle a\rangle X$. Hence, the property we seek is as follows:

$$\mu X.(P \;\wedge\; \nu X.\langle a\rangle X) \;\vee\; \langle a\rangle X.$$

This formula is fine; however, to avoid confusion it is best to use different variables for the two fixed point constructions:

$$\mu X.(P \;\wedge\; \nu Y.\langle a\rangle Y) \;\vee\; \langle a\rangle X.$$

P holds at every state along every a^ω path.

We first express the property that P holds along every (finite or infinite) path of a transitions:

$$\nu X.P \;\wedge\; [a]X.$$

As a greatest fixed point property, the only way this property can fail is if we reach a state after some number of a transitions in which P does not hold. If this is the case, then we want to ensure that we are *not* on an a^ω path; that is, that from this state in which P fails to hold we cannot continue along an a^ω path – which is the second property we considered above: $\mu X.[a]X$. Hence the property we seek is as follows:

$$\nu X.(P \vee \mu X.[a]X) \;\wedge\; [a]X.$$

Again it is sensible to use different variables for the two fixed point constructions:

$$\nu X.(P \vee \mu Y.[a]Y) \;\wedge\; [a]X.$$

Exercise 15.13 (Solution on page 491)

Give mu-calculus formulæ for the following properties. In each case give an intuitive explanation of your solution.

1. **P almost always holds along some a^ω path.**

   Note: To say that something holds *almost always* means *always apart from a finite number of times*. Thus, this property says that P holds everywhere along some a^ω path after some point along this path.

2. **P holds infinitely often along some a^ω path.**

15.7 Additional Exercises

1. Give a semantic definition for the ***weak until*** temporal operator $P \,\mathrm{W}\, Q$ which asserts that the property P remains true until the property Q becomes true, but allows that the property Q may never become true (in which case the property P remains true for as long as the process evolves).

2. We noted in Section 13.2 that we can express the property that the action a *must* happen as:

$$P \;=\; \langle a\rangle\mathsf{true} \;\wedge\; \bigwedge_{b\neq a}[b]\mathsf{false}$$

which says that a may happen and nothing other than a may happen.

What is the difference between the property *"eventually a must happen"* as expressed by the temporal formula $\mathrm{Ev}\, P$ (where the Eventually temporal operator was defined in Exercise 15.3) and the property *"a must eventually happen"*?

(Hint: exactly one of these properties holds for the process $b.\mathbf{0}+a.a.\mathbf{0}$.)

3. Prove Theorem 15.3.
4. Prove that if X is not a free variable of P then for any $S \subseteq \mathsf{States}$, $\|P\|_{\mathsf{V}[X\mapsto S]} = \|P\|_{\mathsf{V}}$.
5. Prove that $E \models_{\mathsf{V}} P$ if, and only if, $E \in \|P\|_{\mathsf{V}}$, where E ranges over formulæ of the modal mu-calculus.
6. Prove the second part of Theorem 15.10, that $f^n(\mathsf{States}) = \|\nu^n X.P\|_{\mathsf{V}}$ where $f(S) = \|P\|_{\mathsf{V}[X\mapsto S]}$.
7. Express the following properties in the modal mu-calculus.
   (a) P holds at some state along every a^ω path.
   (b) P almost always holds along every a^ω paths
   (c) P holds infinitely often along every a^ω path.
8. Express the property of mutual exclusion in the modal mu-calculus; that is, the property that whenever an `entry` action occurs (signifying that a process has entered the critical region), then a further `entry` action cannot occur until an `exit` action occurs.
9. The extended modality $\langle a\rangle^* P$ expresses that the property P holds after some number of a transitions, while the extended modality $[a]^* P$ expresses that the property P holds after any number of a transitions.

   Express these extended modalities as mu-calculus formulæ.
10. Express the following properties in the modal mu-calculus.
    (a) In some run the action a does not happen.
    (b) The actions a and b happen alternately forever (starting with the action a), with any number of occurrences of other actions before and between the a and b actions.
    (c) In any run, a and b happen infinitely often.
    (d) If a and b happen infinitely often, then P is true infinitely often.
    (e) In any run, P is true at least twice.
    (f) In any run, P is true exactly twice.
11. Let $E \overset{\mathrm{def}}{=} a.E + a.F$, $F \overset{\mathrm{def}}{=} b.G$, and $G \overset{\mathrm{def}}{=} a.G$, and consider the following two properties:

$$\Phi_1 \;=\; \mu Y.\Big(\nu X.\langle a\rangle\mathsf{true} \;\wedge\; [-]X\Big) \;\vee\; [-]Y$$

$$\Phi_2 \;=\; \mu Y.\nu X.\Big(\langle a\rangle \mathsf{true} \;\wedge\; [-]X\Big) \;\vee\; [-]Y$$

The process state E satisfies Φ_2 but not Φ_1.

Can you understand and explain why this is the case?

Can you work out what these two properties are expressing?