

2IW05 – Software Specification

Modal Logic - Part II

Mohammad Mousavi

Formal Systems Analysis (FSA)
Model-Driven Software Engineering
Department of Computer Science
TU/Eindhoven

December 19, 2011

Outline

1 Hennessy-Milner logic

2 Semantics of HML

3 Recursion

4 Semantics of Recursion

Syntax of Hennessy-Milner logic

for $a \in Act$

$$F ::= true \mid false \mid \neg F \mid F \wedge F \mid F \vee F \mid \langle a \rangle F \mid [a]F$$

Syntax of Hennessy-Milner logic

for $a \in Act$

$$F ::= true \mid false \mid \neg F \mid F \wedge F \mid F \vee F \mid \langle a \rangle F \mid [a]F$$

where

- $\langle a \rangle F$ denotes that it is possible to perform action a and thereby (in the next state) satisfy F

Syntax of Hennessy-Milner logic

for $a \in Act$

$$F ::= true \mid false \mid \neg F \mid F \wedge F \mid F \vee F \mid \langle a \rangle F \mid [a]F$$

where

- $\langle a \rangle F$ denotes that it is possible to perform action a and thereby (in the next state) satisfy F
- $[a]F$ denotes that no matter how a process performs action a afterwards necessarily F holds

Syntax of Hennessy-Milner logic

for $a \in Act$

$$F ::= true \mid false \mid \neg F \mid F \wedge F \mid F \vee F \mid \langle a \rangle F \mid [a]F$$

where

- $\langle a \rangle F$ denotes that it is possible to perform action a and thereby (in the next state) satisfy F
- $[a]F$ denotes that no matter how a process performs action a afterwards necessarily F holds
- There is a minimal subset

Syntax of Hennessy-Milner logic (recap)

For $A = \{a_1, \dots, a_n\} \subseteq \text{Act}$ with $n \geq 1$

- $\langle A \rangle F$ denotes $\langle a_1 \rangle F \vee \dots \vee \langle a_n \rangle F$ and $\langle \emptyset \rangle F = \text{false}$

Syntax of Hennessy-Milner logic (recap)

For $A = \{a_1, \dots, a_n\} \subseteq Act$ with $n \geq 1$

- $\langle A \rangle F$ denotes $\langle a_1 \rangle F \vee \dots \vee \langle a_n \rangle F$ and $\langle \emptyset \rangle F = false$
- $[A]F$ denotes $[a_1]F \wedge \dots \wedge [a_n]F$ and $[\emptyset]F = true$

Example (recap)

- the scientist can always produce a publication immediately after having drunk two coffees in a row

Example (recap)

- the scientist can always produce a publication immediately after having drunk two coffees in a row

$[coffee][coffee](\langle pub \rangle true \wedge [Act \setminus \{pub\}] false)$

Outline

1 Hennessy-Milner logic

2 Semantics of HML

3 Recursion

4 Semantics of Recursion

Semantics of HML

With each formula associate a set of states where the formula is valid.

$\llbracket F \rrbracket \subseteq S$ is defined inductively by

Semantics of HML

With each formula associate a set of states where the formula is valid.

$\llbracket F \rrbracket \subseteq S$ is defined inductively by

1 $\llbracket true \rrbracket = S$

Semantics of HML

With each formula associate a set of states where the formula is valid.

$\llbracket F \rrbracket \subseteq S$ is defined inductively by

1 $\llbracket true \rrbracket = S$

2 $\llbracket false \rrbracket = \emptyset$

Semantics of HML

With each formula associate a set of states where the formula is valid.

$\llbracket F \rrbracket \subseteq S$ is defined inductively by

1 $\llbracket true \rrbracket = S$

2 $\llbracket false \rrbracket = \emptyset$

3 $\llbracket F \wedge G \rrbracket = \llbracket F \rrbracket \cap \llbracket G \rrbracket$

Semantics of HML

With each formula associate a set of states where the formula is valid.

$\llbracket F \rrbracket \subseteq S$ is defined inductively by

1 $\llbracket \text{true} \rrbracket = S$

2 $\llbracket \text{false} \rrbracket = \emptyset$

3 $\llbracket F \wedge G \rrbracket = \llbracket F \rrbracket \cap \llbracket G \rrbracket$

4 $\llbracket F \vee G \rrbracket = \llbracket F \rrbracket \cup \llbracket G \rrbracket$

Semantics of HML

With each formula associate a set of states where the formula is valid.

$\llbracket F \rrbracket \subseteq S$ is defined inductively by

1 $\llbracket true \rrbracket = S$

2 $\llbracket false \rrbracket = \emptyset$

3 $\llbracket F \wedge G \rrbracket = \llbracket F \rrbracket \cap \llbracket G \rrbracket$

4 $\llbracket F \vee G \rrbracket = \llbracket F \rrbracket \cup \llbracket G \rrbracket$

5 $\llbracket \langle a \rangle F \rrbracket = \langle a \cdot \rangle \llbracket F \rrbracket$

where $\langle a \cdot \rangle$ is defined by

$$\langle a \cdot \rangle T = \{p \in S \mid \exists p'. p \xrightarrow{a} p' \text{ and } p' \in T\}$$

Semantics of HML

With each formula associate a set of states where the formula is valid.

$\llbracket F \rrbracket \subseteq S$ is defined inductively by

1 $\llbracket \text{true} \rrbracket = S$

2 $\llbracket \text{false} \rrbracket = \emptyset$

3 $\llbracket F \wedge G \rrbracket = \llbracket F \rrbracket \cap \llbracket G \rrbracket$

4 $\llbracket F \vee G \rrbracket = \llbracket F \rrbracket \cup \llbracket G \rrbracket$

5 $\llbracket \langle a \rangle F \rrbracket = \langle \cdot a \cdot \rangle \llbracket F \rrbracket$

where $\langle \cdot a \cdot \rangle$ is defined by

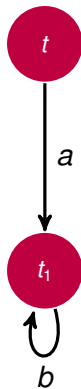
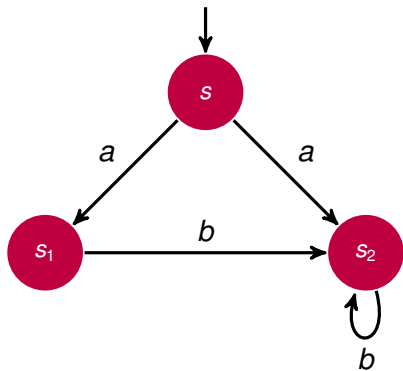
$$\langle \cdot a \cdot \rangle T = \{p \in S \mid \exists p'. p \xrightarrow{a} p' \text{ and } p' \in T\}$$

6 $\llbracket [a] F \rrbracket = [\cdot a \cdot] \llbracket F \rrbracket$

where $[\cdot a \cdot]$ is defined by

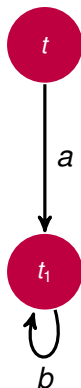
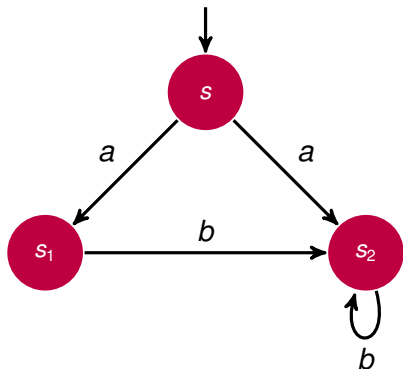
$$[\cdot a \cdot] T = \{p \in S \mid \forall p'. p \xrightarrow{a} p' \Rightarrow p' \in T\}$$

Examples



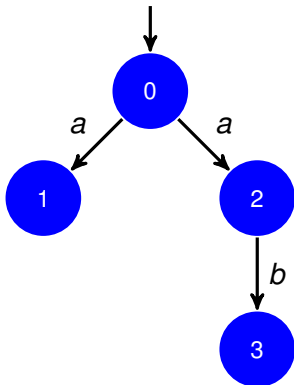
- $\langle \cdot a \cdot \rangle \{s_1, t_1\} = \{s, t\}$

Examples



- $\langle \cdot a \cdot \rangle \{s_1, t_1\} = \{s, t\}$
- $[\cdot a \cdot] \{s_1, t_1\} = \{s_1, s_2, t, t_1\}$

Is the HML formula $\langle a \rangle \langle b \rangle \text{true}$ satisfied by the labeled transition system (i.e., by its **initial state**)?



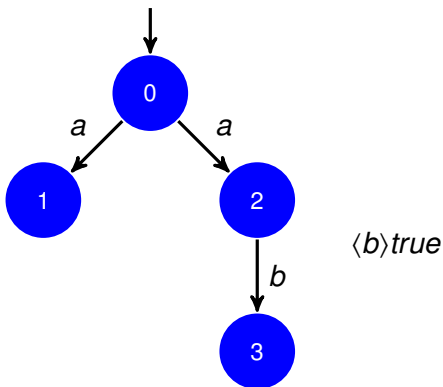
Subformulas

true

$\langle b \rangle \text{true}$

$\langle a \rangle \langle b \rangle \text{true}$

Is the HML formula $\langle a \rangle \langle b \rangle \text{true}$ satisfied by the labeled transition system (i.e., by its **initial state**)?



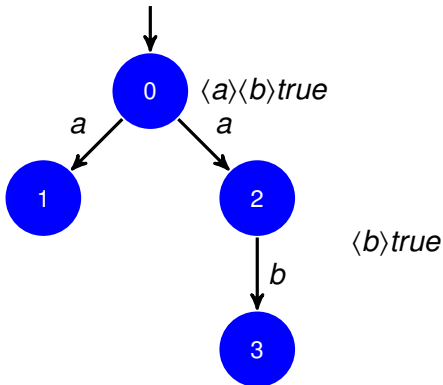
Subformulas

true

$\langle b \rangle \text{true}$

$\langle a \rangle \langle b \rangle \text{true}$

Is the HML formula $\langle a \rangle \langle b \rangle \text{true}$ satisfied by the labeled transition system (i.e., by its **initial state**)?



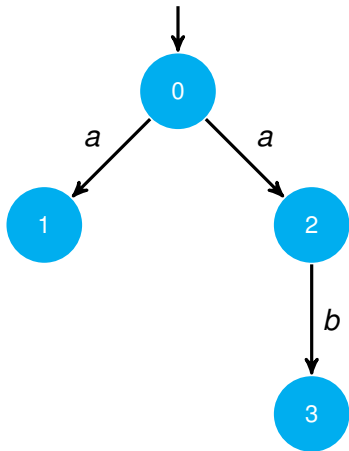
Subformulas

true

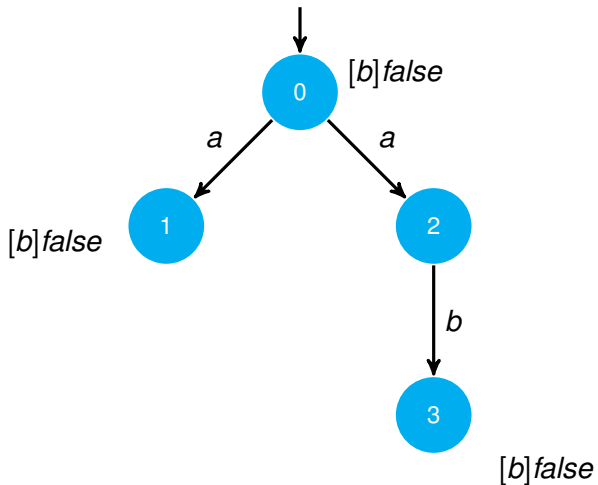
$\langle b \rangle \text{true}$

$\langle a \rangle \langle b \rangle \text{true}$

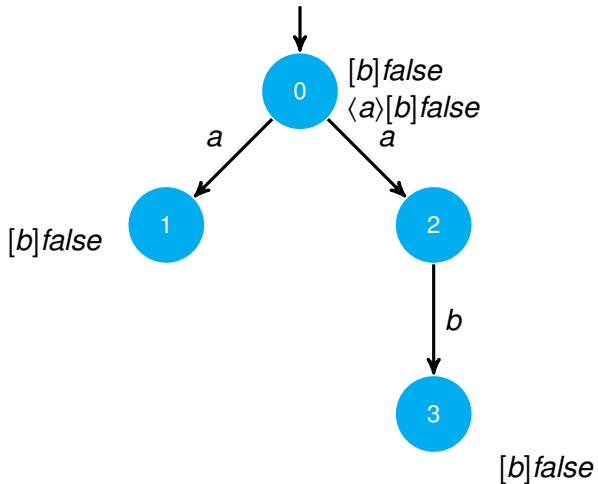
Is the HML formula $\langle a \rangle [b] \text{false}$ satisfied?



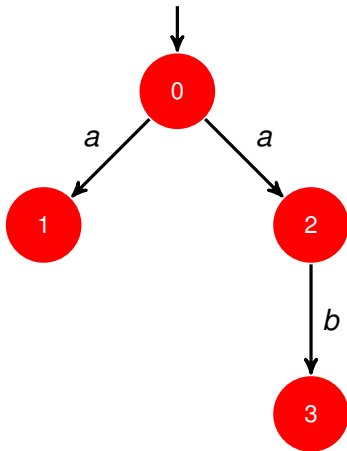
Is the HML formula $\langle a \rangle [b] \text{false}$ satisfied?



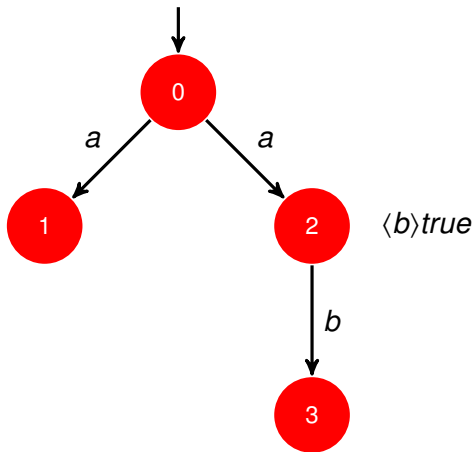
Is the HML formula $\langle a \rangle [b] \text{false}$ satisfied?



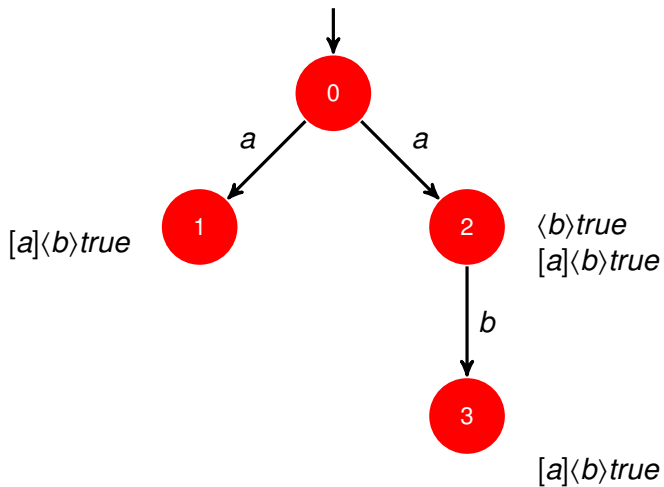
Is the HML formula $[a]\langle b \rangle true$ satisfied?



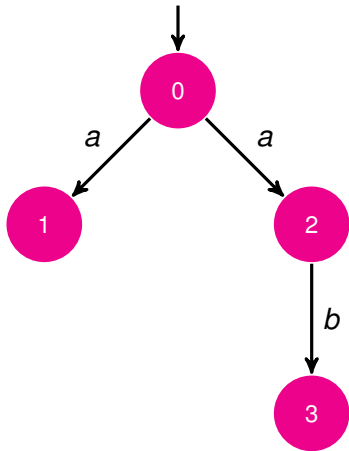
Is the HML formula $[a]\langle b \rangle \text{true}$ satisfied?



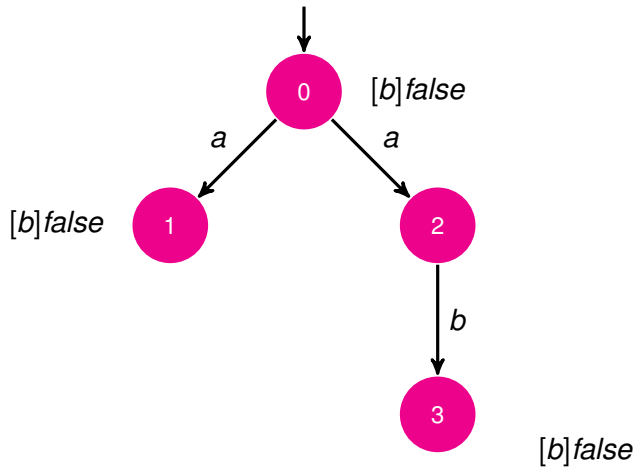
Is the HML formula $[a]\langle b \rangle \text{true}$ satisfied?



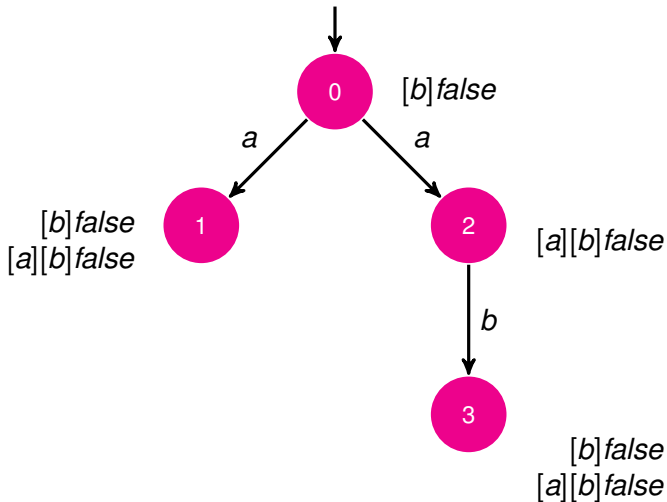
Is the HML formula $[a][b]false$ satisfied?



Is the HML formula $[a][b]false$ satisfied?



Is the HML formula $[a][b]false$ satisfied?



Outline

1 Hennessy-Milner logic

2 Semantics of HML

3 Recursion

4 Semantics of Recursion

Limitations of HML

Limited expressiveness of HML

Using Hennessy-Milner Logic we can only describe properties of behaviors with a finite depth.

Modal depth

- $md(true) = md(false) = 0$
- $md(F \wedge G) = md(F \vee G) = \max\{md(F), md(G)\}$
- $md([a]F) = md(\langle a \rangle F) = md(F) + 1$

Temporal Properties not Expressible in HML

- $Inv(F)$ iff all reachable states satisfy F

$$Inv(F) = F \wedge [Act]F \wedge [Act][Act]F \wedge [Act][Act][Act]F \wedge \dots$$

- $Pos(F)$ iff there is a reachable state which satisfies F

$$Pos(F) = F \vee \langle Act \rangle F \vee \langle Act \rangle \langle Act \rangle F \vee \langle Act \rangle \langle Act \rangle \langle Act \rangle F \vee \dots$$

Temporal Properties not Expressible in HML

- $Inv(F)$ iff all reachable states satisfy F

$$Inv(F) = F \wedge [Act]F \wedge [Act][Act]F \wedge [Act][Act][Act]F \wedge \dots$$

- $Pos(F)$ iff there is a reachable state which satisfies F

$$Pos(F) = F \vee \langle Act \rangle F \vee \langle Act \rangle \langle Act \rangle F \vee \langle Act \rangle \langle Act \rangle \langle Act \rangle F \vee \dots$$

Problems

- infinite formulae are not allowed in HML
- infinite formulae are difficult to handle

Why not to use **recursion**?

- $Inv(F)$ expressed by $X \stackrel{\text{def}}{=} F \wedge [Act]X$
- $Pos(F)$ expressed by $X \stackrel{\text{def}}{=} F \vee \langle Act \rangle X$

Why not to use **recursion**?

- $Inv(F)$ expressed by $X \stackrel{\text{def}}{=} F \wedge [Act]X$
- $Pos(F)$ expressed by $X \stackrel{\text{def}}{=} F \vee \langle Act \rangle X$

Recursion on natural numbers

$$n : n \stackrel{\text{def}}{=} n^2$$

$$n : n \stackrel{\text{def}}{=} n + 1$$

$$n : n \stackrel{\text{def}}{=} 1 \times n$$

HML with one recursively defined variable

Syntax of Formulae

Formulae are given by the following abstract syntax

$$F ::= X \mid true \mid false \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \langle a \rangle F \mid [a]F$$

where $a \in Act$ and X is a distinguished variable with a definition

- $X \stackrel{\min}{=} F_X$, or $X \stackrel{\max}{=} F_X$

such that F_X is a formula of the logic (which can contain X).

HML with one recursively defined variable

Syntax of Formulae

Formulae are given by the following abstract syntax

$$F ::= X \mid true \mid false \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \langle a \rangle F \mid [a]F$$

where $a \in Act$ and X is a distinguished variable with a definition

- $X \stackrel{\min}{=} F_X$, or $X \stackrel{\max}{=} F_X$

such that F_X is a formula of the logic (which can contain X).

Alternative syntax:

$$F ::= X \mid true \mid false \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid \langle a \rangle F \mid [a]F \\ \mid \mu X.F \mid \nu X.F$$

Example:

$$X^{\min} \equiv X$$

Any set of states S satisfies the set-equation $X = X$. The least such set is \emptyset .

Example:

$$X \stackrel{\text{min}}{=} X$$

Any set of states S satisfies the set-equation $X = X$. The least such set is \emptyset .

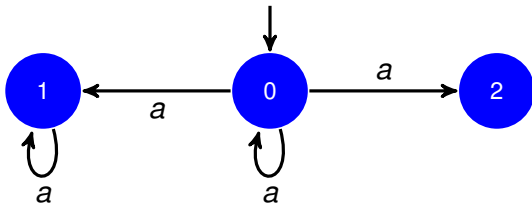
Example:

$$X \stackrel{\text{max}}{=} X$$

Any set of states S satisfies the set-equation $X = X$. The greatest such set is S .

Eventually 'a' will be disabled:

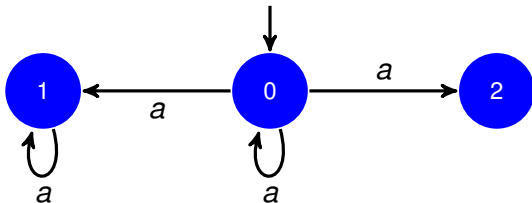
$$X \stackrel{?}{=} [a]false \vee \langle Act \rangle X$$



The property is valid for the labeled transition system

Eventually 'a' will be disabled:

$$X \stackrel{?}{=} [a]false \vee \langle Act \rangle X$$

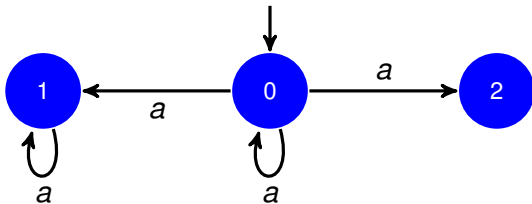


The property is valid for the labeled transition system

Solutions of this equation are the sets: {0, 2} and {0, 1, 2}

Eventually 'a' will be disabled:

$$X \stackrel{?}{=} [a]false \vee \langle Act \rangle X$$



The property is valid for the labeled transition system

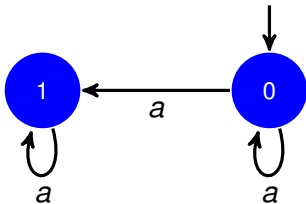
Solutions of this equation are the sets: {0, 2} and {0, 1, 2}

We intended to describe the least solution!

$$X \stackrel{\min}{=} [a]false \vee \langle Act \rangle X$$

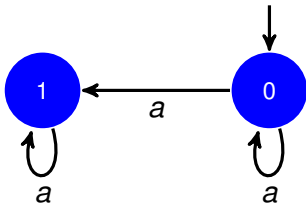
Example: A state can be reached where a cannot be executed:

$$X \stackrel{\text{min}}{=} [a] \text{false} \vee \langle \text{Act} \rangle X$$



Example: A state can be reached where a cannot be executed:

$$X \stackrel{\text{min}}{=} [a] \text{false} \vee \langle \text{Act} \rangle X$$

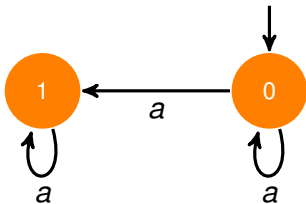


The unique least solution for this equation is the set of states \emptyset

Hence the property is not valid for the labeled transition system

Example: In every reachable state an a -transition is possible

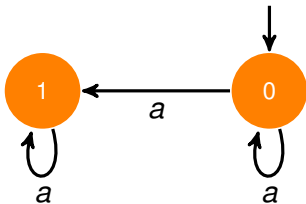
$$X \stackrel{?}{=} \langle a \rangle true \wedge [Act]X$$



Solutions: \emptyset , $\{1\}$, and $\{0, 1\}$

Example: In every reachable state an a -transition is possible

$$X \stackrel{?}{=} \langle a \rangle true \wedge [Act]X$$



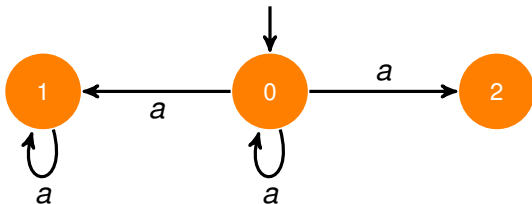
Solutions: \emptyset , $\{1\}$, and $\{0, 1\}$

We intended to describe the greatest solution!

$$X \stackrel{\max}{=} \langle a \rangle true \wedge [Act]X$$

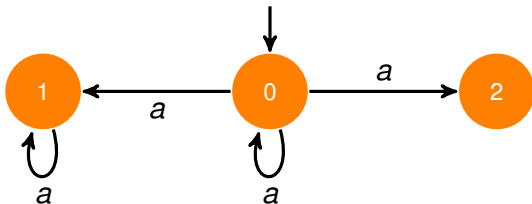
Example: In every reachable state an a -transition is possible

$$X \stackrel{\text{max}}{=} \langle a \rangle \text{true} \wedge [\text{Act}]X$$



Example: In every reachable state an a -transition is possible

$$X \stackrel{\max}{=} \langle a \rangle \text{true} \wedge [\text{Act}]X$$

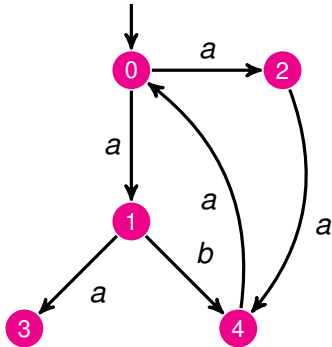


The greatest solution for this equation is the set of states {1}

Thus property is not valid for the labeled transition system

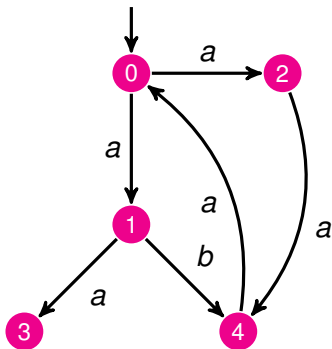
$$X \stackrel{\min}{=} \langle b \rangle \text{true} \vee \langle \text{Act} \rangle X$$

There is a path to a state where a b is possible



$$X \stackrel{\min}{=} \langle b \rangle \text{true} \vee \langle \text{Act} \rangle X$$

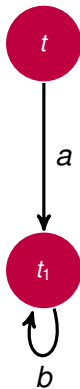
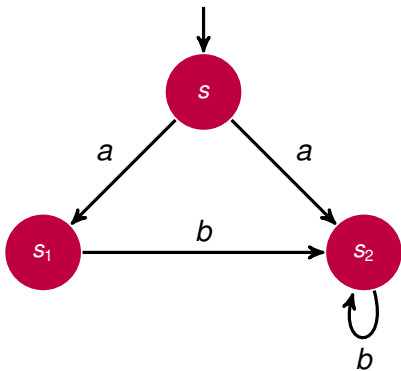
There is a path to a state where a b is possible



The least solution is the set of states $\{0, 1, 2, 4\}$; thus, property is valid for the labeled transition system

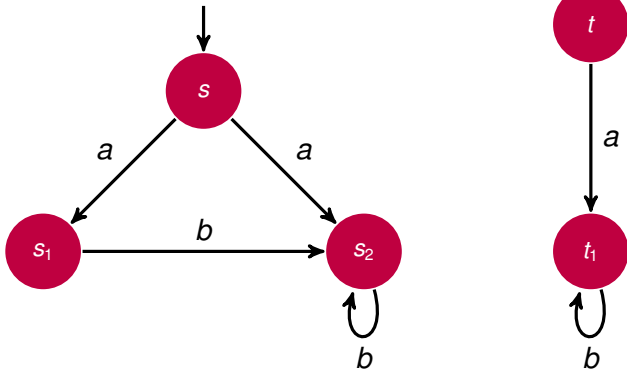
$$X \stackrel{\text{max}}{=} \langle b \rangle \text{true} \wedge [b] X$$

initially and after each b , one can take a b transition



$$X \stackrel{\text{max}}{=} \langle b \rangle \text{true} \wedge [b]X$$

initially and after each b , one can take a b transition



The greatest solution is the set of states $\{s_1, s_2, t_1\}$.

Formulas for the properties that cannot be expressed in HML

- the scientist never drinks beer

$$X \stackrel{\text{max}}{=} [\text{beer}] \text{false} \wedge [\text{Act}] X$$

Formulas for the properties that cannot be expressed in HML

- the scientist never drinks beer

$$X \stackrel{\text{max}}{=} [\text{beer}] \text{false} \wedge [\text{Act}] X$$

- the scientist always produces a publication after drinking coffee

$$X \stackrel{\text{max}}{=} [\text{coffee}] (\langle \text{pub} \rangle \text{true} \wedge [\text{Act} \setminus \{\text{pub}\}] \text{false}) \wedge [\text{Act}] X$$

Formulas for the properties that cannot be expressed in HML

- the scientist never drinks beer

$$X \stackrel{\text{max}}{=} [\text{beer}] \text{false} \wedge [\text{Act}] X$$

- the scientist always produces a publication after drinking coffee

$$X \stackrel{\text{max}}{=} [\text{coffee}] (\langle \text{pub} \rangle \text{true} \wedge [\text{Act} \setminus \{\text{pub}\}] \text{false}) \wedge [\text{Act}] X$$

- $\text{Inv}(F)$

$$X \stackrel{\text{max}}{=} F \wedge [\text{Act}] X$$

Formulas for the properties that cannot be expressed in HML

- the scientist never drinks beer

$$X \stackrel{\text{max}}{=} [\text{beer}] \text{false} \wedge [\text{Act}] X$$

- the scientist always produces a publication after drinking coffee

$$X \stackrel{\text{max}}{=} [\text{coffee}] (\langle \text{pub} \rangle \text{true} \wedge [\text{Act} \setminus \{\text{pub}\}] \text{false}) \wedge [\text{Act}] X$$

- $\text{Inv}(F)$

$$X \stackrel{\text{max}}{=} F \wedge [\text{Act}] X$$

- $\text{Pos}(F)$

$$X \stackrel{\text{min}}{=} F \vee \langle \text{Act} \rangle X$$

Outline

1 Hennessy-Milner logic

2 Semantics of HML

3 Recursion

4 Semantics of Recursion

Semantics of Recursion (one variable)

- With each formula associate a set of states for which it is satisfied

$$\llbracket F \rrbracket \subseteq S$$

Semantics of Recursion (one variable)

- With each formula associate a set of states for which it is satisfied

$$\llbracket F \rrbracket \subseteq S$$

- How to deal with recursion variable X ?

Semantics of Recursion (one variable)

- With each formula associate a set of states for which it is satisfied

$$\llbracket F \rrbracket \subseteq S$$

- How to deal with recursion variable X ?
- Make an assumption on states satisfied by X . For every formula F we define a function $O_F : 2^S \rightarrow 2^S$ s.t.
 - if S is the set of processes that satisfy X
 - then $O_F(S)$ is the set of processes that satisfy F .

Definition of $O_F : 2^S \rightarrow 2^S$

For $S \subseteq S$

$$O_X(T) = T$$

$$O_{true}(T) = S$$

$$O_{false}(T) = \emptyset$$

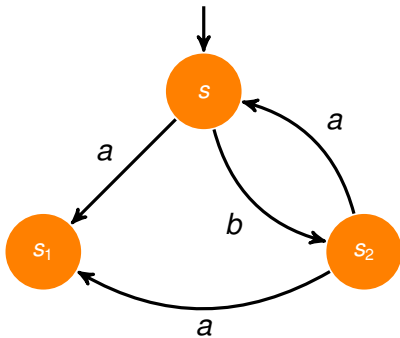
$$O_{F_1 \wedge F_2}(T) = O_{F_1}(T) \cap O_{F_2}(T)$$

$$O_{F_1 \vee F_2}(T) = O_{F_1}(S) \cup O_{F_2}(T)$$

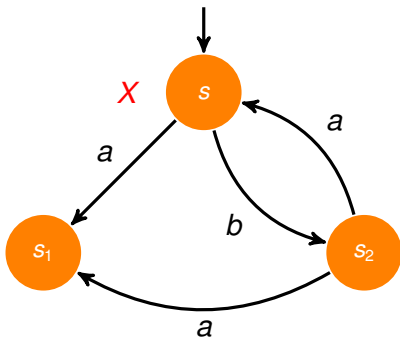
$$O_{\langle a \rangle F}(T) = \langle \cdot a \cdot \rangle O_F(T)$$

$$O_{[a]F}(T) = [\cdot a \cdot] O_F(T)$$

Example

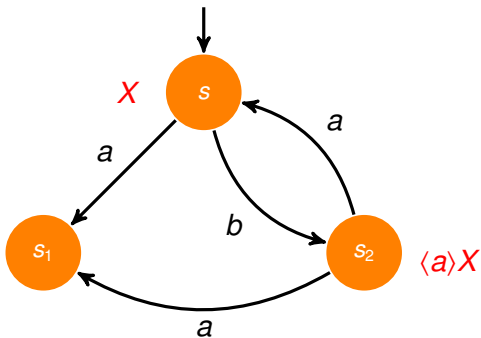


Example



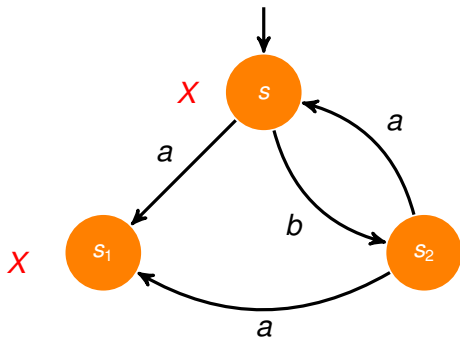
1 $O_{\langle a \rangle X}(\{s\}) = \langle \cdot a \cdot \rangle O_X(\{s\}) = \langle \cdot a \cdot \rangle \{s\} = \{s_2\}$

Example



1 $O_{\langle a \rangle X}(\{s\}) = \langle \cdot a \cdot \rangle O_X(\{s\}) = \langle \cdot a \cdot \rangle \{s\} = \{s_2\}$

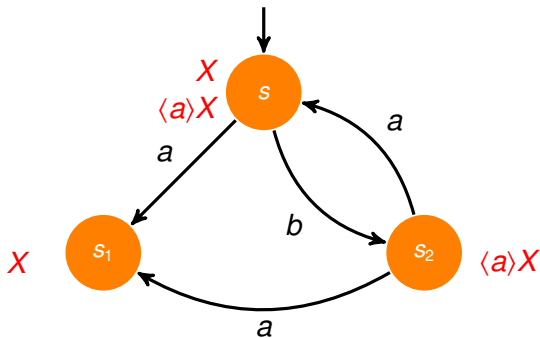
Example



$$1 \quad O_{\langle a \rangle X}(\{s\}) = \langle \cdot a \cdot \rangle O_X(\{s\}) = \langle \cdot a \cdot \rangle \{s\} = \{s_2\}$$

$$2 \quad O_{\langle a \rangle X}(\{s, s_1\}) = \langle \cdot a \cdot \rangle O_X(\{s, s_1\}) = \langle \cdot a \cdot \rangle \{s, s_1\} = \{s, s_2\}$$

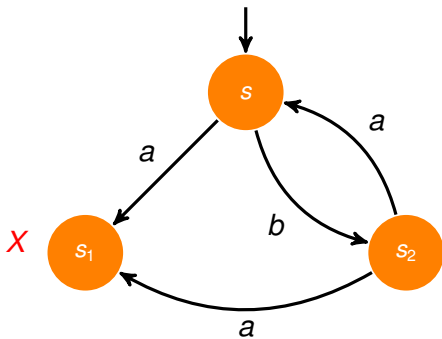
Example



1 $O_{\langle a \rangle X}(\{s\}) = \langle \cdot a \cdot \rangle O_X(\{s\}) = \langle \cdot a \cdot \rangle \{s\} = \{s_2\}$

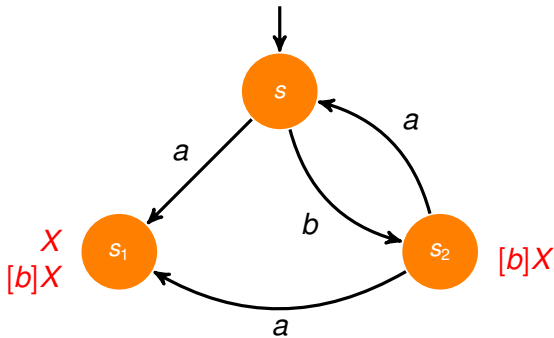
2 $O_{\langle a \rangle X}(\{s, s_1\}) = \langle \cdot a \cdot \rangle O_X(\{s, s_1\}) = \langle \cdot a \cdot \rangle \{s, s_1\} = \{s, s_2\}$

Example



- 1 $O_{\langle a \rangle X}(\{s\}) = \langle \cdot a \cdot \rangle O_X(\{s\}) = \langle \cdot a \cdot \rangle \{s\} = \{s_2\}$
- 2 $O_{\langle a \rangle X}(\{s, s_1\}) = \langle \cdot a \cdot \rangle O_X(\{s, s_1\}) = \langle \cdot a \cdot \rangle \{s, s_1\} = \{s, s_2\}$
- 3 $O_{[b] X}(\{s_1\}) = [\cdot b \cdot] O_X(\{s_1\}) = [\cdot b \cdot] \{s_1\} = \{s_1, s_2\}$

Example



- 1 $O_{\langle a \rangle X}(\{s\}) = \langle \cdot a \cdot \rangle O_X(\{s\}) = \langle \cdot a \cdot \rangle \{s\} = \{s_2\}$
- 2 $O_{\langle a \rangle X}(\{s, s_1\}) = \langle \cdot a \cdot \rangle O_X(\{s, s_1\}) = \langle \cdot a \cdot \rangle \{s, s_1\} = \{s, s_2\}$
- 3 $O_{[b] X}(\{s_1\}) = [\cdot b \cdot] O_X(\{s_1\}) = [\cdot b \cdot] \{s_1\} = \{s_1, s_2\}$

Observation

Semantics of formula F

Observation

Semantics of formula F

- 1 $\llbracket true \rrbracket = S$
- 2 $\llbracket false \rrbracket = \emptyset$
- 3 $\llbracket F \wedge G \rrbracket = \llbracket F \rrbracket \cap \llbracket G \rrbracket$
- 4 $\llbracket F \vee G \rrbracket = \llbracket F \rrbracket \cup \llbracket G \rrbracket$
- 5 $\llbracket \langle a \rangle F \rrbracket = \langle \cdot a \cdot \rrbracket \llbracket F \rrbracket$ where $\langle \cdot a \cdot \rangle : 2^S \rightarrow 2^S$ is defined by

$$\langle \cdot a \cdot \rangle S = \{p \in S \mid \exists p'. p \xrightarrow{a} p' \text{ and } p' \in S\}$$

- 6 $\llbracket [a] F \rrbracket = [\cdot a \cdot] \llbracket F \rrbracket$ where $[\cdot a \cdot] : 2^S \rightarrow 2^S$ is defined by

$$[\cdot a \cdot] S = \{p \in S \mid \forall p'. p \xrightarrow{a} p' \Rightarrow p' \in S\}$$

Observation

Semantics of formula F

- 1 $\llbracket \text{true} \rrbracket = S$
- 2 $\llbracket \text{false} \rrbracket = \emptyset$
- 3 $\llbracket F \wedge G \rrbracket = \llbracket F \rrbracket \cap \llbracket G \rrbracket$
- 4 $\llbracket F \vee G \rrbracket = \llbracket F \rrbracket \cup \llbracket G \rrbracket$
- 5 $\llbracket \langle a \rangle F \rrbracket = \langle \cdot a \cdot \rangle \llbracket F \rrbracket$ where $\langle \cdot a \cdot \rangle : 2^S \rightarrow 2^S$ is defined by

$$\langle \cdot a \cdot \rangle S = \{p \in S \mid \exists p'. p \xrightarrow{a} p' \text{ and } p' \in S\}$$

- 6 $\llbracket [a] F \rrbracket = [\cdot a \cdot] \llbracket F \rrbracket$ where $[\cdot a \cdot] : 2^S \rightarrow 2^S$ is defined by

$$[\cdot a \cdot] S = \{p \in S \mid \forall p'. p \xrightarrow{a} p' \Rightarrow p' \in S\}$$

- 7 If $X \stackrel{\text{min}}{=} F_X$ then $\llbracket X \rrbracket = \bigcap \{S \subseteq S \mid S = O_{F_X}(S)\}$

Observation

Semantics of formula F

- 1 $\llbracket true \rrbracket = S$
- 2 $\llbracket false \rrbracket = \emptyset$
- 3 $\llbracket F \wedge G \rrbracket = \llbracket F \rrbracket \cap \llbracket G \rrbracket$
- 4 $\llbracket F \vee G \rrbracket = \llbracket F \rrbracket \cup \llbracket G \rrbracket$
- 5 $\llbracket \langle a \rangle F \rrbracket = \langle a \cdot \rangle \llbracket F \rrbracket$ where $\langle a \cdot \rangle : 2^S \rightarrow 2^S$ is defined by

$$\langle a \cdot \rangle S = \{p \in S \mid \exists p'. p \xrightarrow{a} p' \text{ and } p' \in S\}$$

- 6 $\llbracket [a] F \rrbracket = [a \cdot] \llbracket F \rrbracket$ where $[a \cdot] : 2^S \rightarrow 2^S$ is defined by

$$[a \cdot] S = \{p \in S \mid \forall p'. p \xrightarrow{a} p' \Rightarrow p' \in S\}$$

- 7 If $X \stackrel{\min}{=} F_X$ then $\llbracket X \rrbracket = \bigcap \{S \subseteq S \mid S = O_{F_X}(S)\}$
- 8 If $X \stackrel{\max}{=} F_X$ then $\llbracket X \rrbracket = \bigcup \{S \subseteq S \mid S = O_{F_X}(S)\}$

Let S be a finite set.

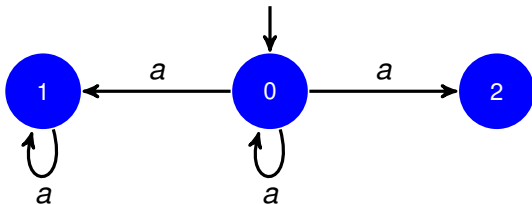
Computing the solution of $X \stackrel{\min}{=} F_X$

There exists a natural number $m > 0$ such that $\llbracket X \rrbracket = O_{F_X}^m(\emptyset)$

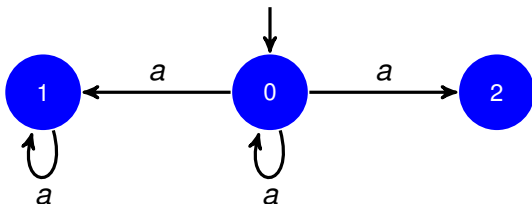
Computing the solution of $X \stackrel{\max}{=} F_X$

There exist a natural number $M > 0$ such that $\llbracket X \rrbracket = O_{F_X}^M(S)$

Example: $X \stackrel{\text{min}}{\equiv} [a] \text{false} \vee \langle \text{Act} \rangle X$

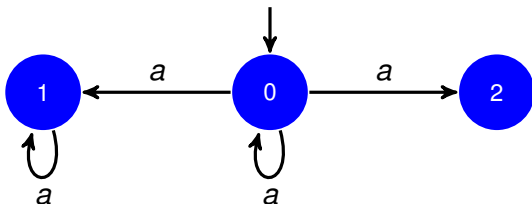


Example: $X \stackrel{\min}{=} [a]false \vee \langle Act \rangle X$



$$\begin{aligned} O_{F_X}(S) &= O_{[a]false}(S) \cup O_{\langle Act \rangle X}(S) \\ &= [\cdot a \cdot] O_{false}(S) \cup \langle \cdot Act \cdot \rangle O_X(S) \\ &= [\cdot a \cdot] \emptyset \cup \langle \cdot Act \cdot \rangle S \\ &= \{2\} \cup \langle \cdot Act \cdot \rangle S \end{aligned}$$

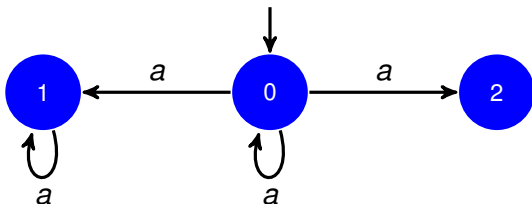
Example: $X \stackrel{\min}{=} [a]false \vee \langle Act \rangle X$



$$\begin{aligned} O_{F_X}(S) &= O_{[a]false}(S) \cup O_{\langle Act \rangle X}(S) \\ &= [\cdot a \cdot] O_{false}(S) \cup \langle \cdot Act \cdot \rangle O_X(S) \\ &= [\cdot a \cdot] \emptyset \cup \langle \cdot Act \cdot \rangle S \\ &= \{2\} \cup \langle \cdot Act \cdot \rangle S \end{aligned}$$

① $O_{F_X}(\emptyset) = \{2\} \cup \langle \cdot Act \cdot \rangle \emptyset = \{2\} \cup \emptyset = \{2\}$

Example: $X \stackrel{\min}{=} [a]false \vee \langle Act \rangle X$

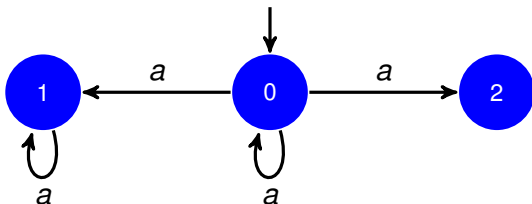


$$\begin{aligned} O_{F_X}(S) &= O_{[a]false}(S) \cup O_{\langle Act \rangle X}(S) \\ &= [\cdot a \cdot] O_{false}(S) \cup \langle \cdot Act \cdot \rangle O_X(S) \\ &= [\cdot a \cdot] \emptyset \cup \langle \cdot Act \cdot \rangle S \\ &= \{2\} \cup \langle \cdot Act \cdot \rangle S \end{aligned}$$

① $O_{F_X}(\emptyset) = \{2\} \cup \langle \cdot Act \cdot \rangle \emptyset = \{2\} \cup \emptyset = \{2\}$

② $O_{F_X}(\{2\}) = \{2\} \cup \langle \cdot Act \cdot \rangle \{2\} = \{2\} \cup \{0\} = \{0, 2\}$

Example: $X \stackrel{\min}{=} [a]false \vee \langle Act \rangle X$



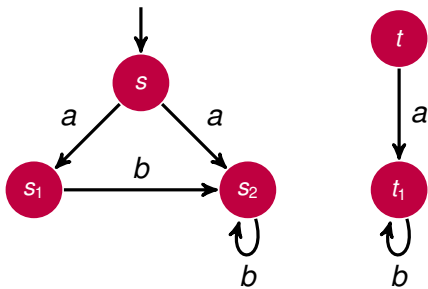
$$\begin{aligned} O_{F_X}(S) &= O_{[a]false}(S) \cup O_{\langle Act \rangle X}(S) \\ &= [\cdot a \cdot] O_{false}(S) \cup \langle \cdot Act \cdot \rangle O_X(S) \\ &= [\cdot a \cdot] \emptyset \cup \langle \cdot Act \cdot \rangle S \\ &= \{2\} \cup \langle \cdot Act \cdot \rangle S \end{aligned}$$

① $O_{F_X}(\emptyset) = \{2\} \cup \langle \cdot Act \cdot \rangle \emptyset = \{2\} \cup \emptyset = \{2\}$

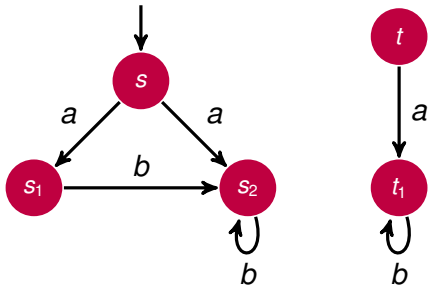
② $O_{F_X}(\{2\}) = \{2\} \cup \langle \cdot Act \cdot \rangle \{2\} = \{2\} \cup \{0\} = \{0, 2\}$

③ $O_{F_X}(\{0, 2\}) = \{2\} \cup \langle \cdot Act \cdot \rangle \{0, 2\} = \{2\} \cup \{0\} = \{0, 2\}$

Example: $X \stackrel{\text{max}}{\equiv} \langle b \rangle \text{true} \wedge [b]X$

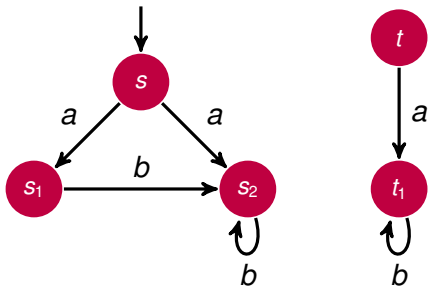


Example: $X \stackrel{\max}{=} \langle b \rangle \text{true} \wedge [b]X$

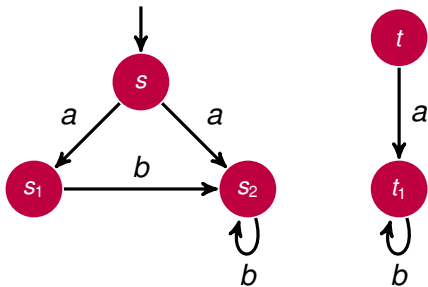


$$\begin{aligned} O_{F_X}(S) &= O_{\langle b \rangle \text{true}}(S) \cap O_{[b]X}(S) \\ &= \langle \cdot b \cdot \rangle O_{\text{true}}(S) \cap [\cdot b \cdot] O_X(S) \\ &= \langle \cdot b \cdot \rangle S \cap [\cdot b \cdot] S \\ &= \{s_1, s_2, t_1\} \cap [\cdot b \cdot] S \end{aligned}$$

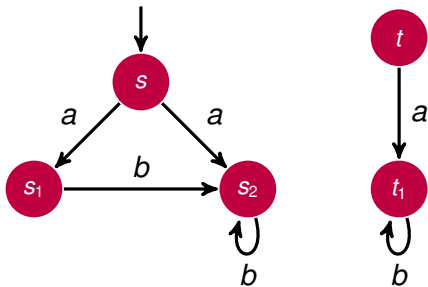
Example: $X \stackrel{\text{max}}{\equiv} \langle b \rangle \text{true} \wedge [b]X$



Example: $X \stackrel{\text{max}}{\equiv} \langle b \rangle \text{true} \wedge [b]X$

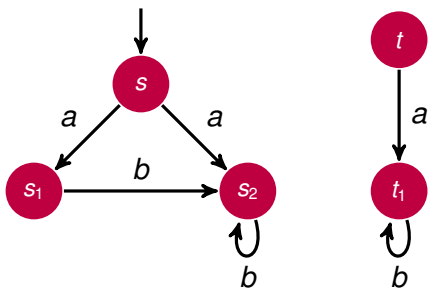


Example: $X \stackrel{\text{max}}{=} \langle b \rangle \text{true} \wedge [b]X$



1 $O_{F_X}(S) = \{s_1, s_2, t_1\} \cap [\cdot b]S = \{s_1, s_2, t_1\} \cap \{s, s_1, s_2, t, t_1\} = \{s_1, s_2, t_1\}$

Example: $X \stackrel{\text{max}}{=} \langle b \rangle \text{true} \wedge [b]X$



- 1 $O_{F_X}(S) = \{s_1, s_2, t_1\} \cap [\cdot b \cdot]S = \{s_1, s_2, t_1\} \cap \{s, s_1, s_2, t, t_1\} = \{s_1, s_2, t_1\}$
- 2 $O_{F_X}(\{s_1, s_2, t_1\}) = \{s_1, s_2, t_1\} \cap [\cdot b \cdot]\{s_1, s_2, t_1\} = \{s_1, s_2, t_1\} \cap \{s, s_1, s_2, t, t_1\} = \{s_1, s_2, t_1\}$

Some temporal properties

- *Safe(F)*: for some execution F holds everywhere

$$X \stackrel{\text{max}}{=} F \wedge ([Act]false \vee \langle Act \rangle X)$$

Some temporal properties

- *Safe(F)*: for some execution F holds everywhere

$$X \stackrel{\text{max}}{\equiv} F \wedge ([Act]false \vee \langle Act \rangle X)$$

- *Even(F)*: eventually F will hold (in every execution)

$$X \stackrel{\text{min}}{\equiv} F \vee (\langle Act \rangle true \wedge [Act]X)$$

Some temporal properties

- *Safe(F)*: for some execution F holds everywhere

$$X \stackrel{\text{max}}{=} F \wedge ([Act]false \vee \langle Act \rangle X)$$

- *Even(F)*: eventually F will hold (in every execution)

$$X \stackrel{\text{min}}{=} F \vee (\langle Act \rangle true \wedge [Act]X)$$

- $F \mathcal{U}^w G$: F holds in all states until a state is reached where G holds

$$X \stackrel{\text{max}}{=} G \vee (F \wedge [Act]X)$$

Some temporal properties

- *Safe(F)*: for some execution F holds everywhere

$$X \stackrel{\text{max}}{\equiv} F \wedge ([\text{Act}] \text{false} \vee \langle \text{Act} \rangle X)$$

- *Even(F)*: eventually F will hold (in every execution)

$$X \stackrel{\text{min}}{\equiv} F \vee (\langle \text{Act} \rangle \text{true} \wedge [\text{Act}] X)$$

- $F \mathcal{U}^w G$: F holds in all states until a state is reached where G holds

$$X \stackrel{\text{max}}{\equiv} G \vee (F \wedge [\text{Act}] X)$$

- $F \mathcal{U}^s G$: sooner or later G holds and until then F holds in all states traversed

$$X \stackrel{\text{min}}{\equiv} G \vee (F \wedge \langle \text{Act} \rangle \text{true} \wedge [\text{Act}] X)$$

Some temporal properties

Using until we can express e.g. $Inv(F)$ and $Even(F)$:

$Inv(F)$ and $F \mathcal{U}^w false$ are logically equivalent

$Even(F)$ and $true \mathcal{U}^s F$ are logically equivalent