CS228 : Logic for Computer Science

Module 5: Temporal Logic

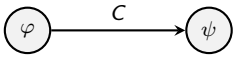
Lecture 23: Linear-time Temporal Logic (LTL)

Instructor: S. Akshay

IIT Bombay, India

Recap: Why temporal logic?

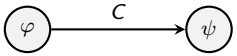
Hoare logic



a program starts and *halts*:
we compare before and after

Recap: Why temporal logic?

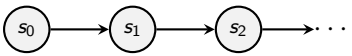
Hoare logic



a program starts and *halts*:
we compare before and after



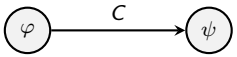
Temporal logic



an operating system, a lift controller, a pacemaker never stop:
we talk about the whole *path* of states

Recap: Why temporal logic?

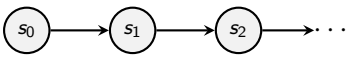
Hoare logic



a program starts and *halts*:
we compare before and after



Temporal logic



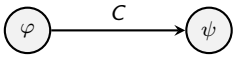
an operating system, a lift controller, a pacemaker never stop:
we talk about the whole *path* of states

Models become dynamic

- Propositional logic and FOL: models are static, one snapshot (a *valuation*, a *structure*) in which a formula is simply true or false, once and for all.
  E.g., $\{p, q\} \not\models p \wedge \neg q$; $(\mathbb{N}, <) \models \forall x \exists y\ x < y$ but $(\{1, 2, 3\}, <) \not\models \forall x \exists y\ x < y$

Recap: Why temporal logic?

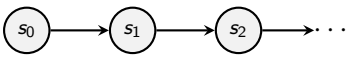
Hoare logic



a program starts and *halts*:
we compare before and after



Temporal logic



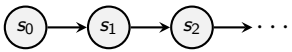
an operating system, a lift controller, a pacemaker never stop:
we talk about the whole *path* of states

Models become dynamic

- Propositional logic and FOL: models are static, one snapshot (a *valuation*, a *structure*) in which a formula is simply true or false, once and for all.
  E.g., $\{p, q\} \not\models p \wedge \neg q$; $(\mathbb{N}, <) \models \forall x \exists y\ x < y$ but $(\{1, 2, 3\}, <) \not\models \forall x \exists y\ x < y$
- Temporal logic: models are dynamic, a graph of states and transitions, and the truth of a formula depends on the state we are in: $M, s \models \varphi$

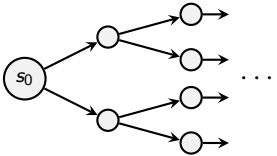
Recap: Why temporal logic?

Linear-time



look along *one path*:
one future at each moment

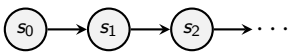
Branching-time



view time as a tree, rooted at present:
many possible futures

Recap: Why temporal logic?

Linear-time

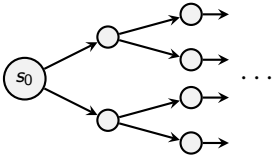


look along *one path*:
one future at each moment



Loki: the Sacred Timeline,
the TVA prunes every branch

Branching-time



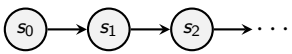
view time as a tree, rooted at present:
many possible futures



Doctor Strange: millions of futures,
only one where we win

Recap: Why temporal logic?

Linear-time $\rightarrow$ LTL

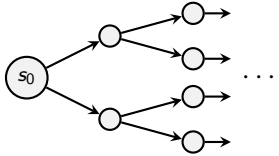


look along *one path*:
one future at each moment



Loki: the Sacred Timeline,
the TVA prunes every branch

Branching-time $\rightarrow$ CTL



view time as a tree, rooted at present:
many possible futures



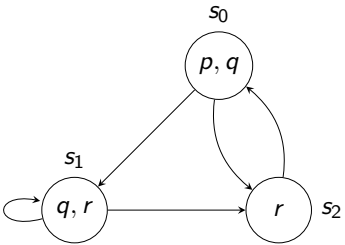
Doctor Strange: millions of futures,
only one where we win

Topic 23.1

Transition Systems

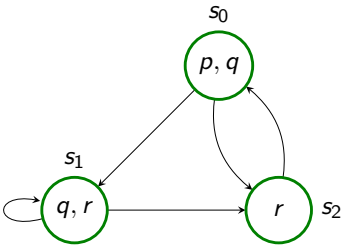
Transition systems as models

To reason about temporal properties of systems, we need a richer model...



A directed graph: each node (state) contains the atomic propositions true in that state

Transition systems as models

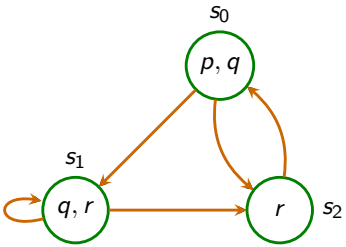


A directed graph: each node (state) contains the atomic propositions true in that state

A Transition system (also called a *Kripke structure*) is a triple $M = (S, \rightarrow, L)$ where,

- S is a set of states

Transition systems as models

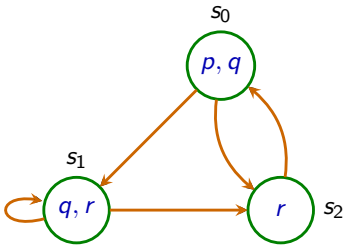


A directed graph: each node (state) contains the atomic propositions true in that state

A Transition system (also called a *Kripke structure*) is a triple $M = (S, \rightarrow, L)$ where,

- S is a set of states
- $\rightarrow$ is a binary relation on S called transition relation s.t., $\forall s \in S, \exists s' \in S$ with $s \rightarrow s'$.

Transition systems as models

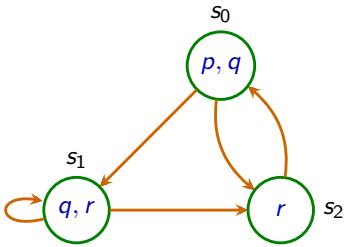


A directed graph: each node (state) contains the atomic propositions true in that state

A Transition system (also called a *Kripke structure*) is a triple $M = (S, \rightarrow, L)$ where,

- S is a set of states
- $\rightarrow$ is a binary relation on S called transition relation s.t., $\forall s \in S, \exists s' \in S$ with $s \rightarrow s'$.
- $L : S \rightarrow \mathcal{P}(AP)$ where
  - AP is a set of atomic propositions: statements about system, e.g., *Printer is busy*
  - 2^{AP} is its power set: the set of all subsets.

Transition systems as models



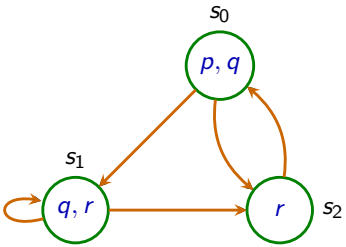
A directed graph: each node (state) contains the atomic propositions true in that state

L is like a valuation: in propositional logic, $m \subseteq AP$ assigned truth values to all propositions; now each state has its own $L(s) \subseteq AP$.

A Transition system (also called a *Kripke structure*) is a triple $M = (S, \rightarrow, L)$ where,

- S is a set of states
- $\rightarrow$ is a binary relation on S called transition relation s.t., $\forall s \in S, \exists s' \in S$ with $s \rightarrow s'$.
- $L : S \rightarrow \mathcal{P}(AP)$ where
  - AP is a set of atomic propositions: statements about system, e.g., *Printer is busy*
  - 2^{AP} is its power set: the set of all subsets.

Transition systems as models



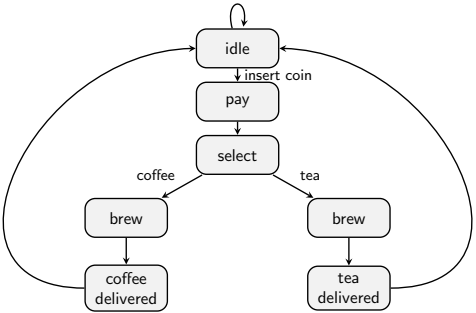
A directed graph: each node (state) contains the atomic propositions true in that state

Exercise 23.1

1. Let $AP = \{p, q, r\}$. List all elements of $\mathcal{P}(AP)$.
2. Write the formal description of M in the diagram.
3. Can M have a deadlocked state, a state from which there is no outgoing transition?
4. If we remove the deadlock condition from the definition, does it matter?

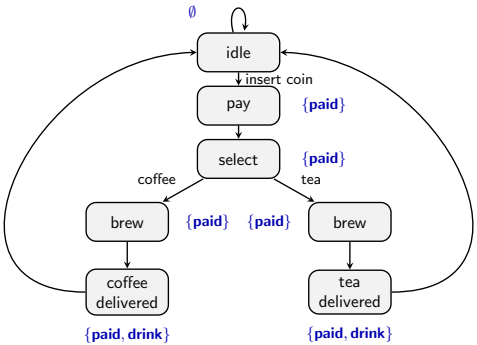
A real transition system: a vending machine

"A vending machine takes a coin, brews coffee or tea, then delivers it."



A real transition system: a vending machine

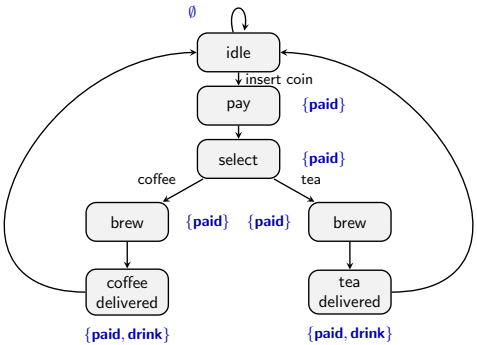
"A vending machine takes a coin, brews coffee or tea, then delivers it."



Atomic propositions $AP = \{\textbf{paid}, \textbf{drink}\}$: the user has paid; a drink is delivered.

A real transition system: a vending machine

"A vending machine takes a coin, brews coffee or tea, then delivers it."



Atomic propositions $AP = \{\textbf{paid}, \textbf{drink}\}$: the user has paid; a drink is delivered.

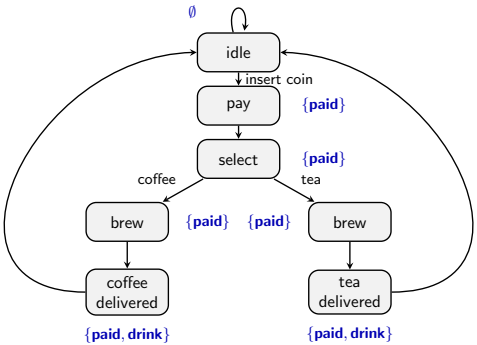
Quick questions

1. Which states can follow *select*?

2. What is the label of *idle*?

A real transition system: a vending machine

"A vending machine takes a coin, brews coffee or tea, then delivers it."



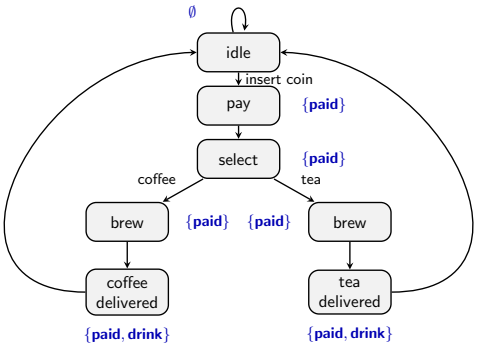
Atomic propositions $AP = \{\textbf{paid}, \textbf{drink}\}$: the user has paid; a drink is delivered.

Quick questions

1. Which states can follow *select*? two different states, both called *brew*: one for coffee, one for tea!
2. What is the label of *idle*? $\emptyset$: no atomic proposition is true there, a legal label.

A real transition system: a vending machine

"A vending machine takes a coin, brews coffee or tea, then delivers it."



Atomic propositions $AP = \{\textbf{paid}, \textbf{drink}\}$: the user has paid; a drink is delivered.

Quick questions

1. Which states can follow *select*? two different states, both called *brew*: one for coffee, one for tea!
2. What is the label of *idle*? $\emptyset$: no atomic proposition is true there, a legal label.

Which properties of this machine would we like to state?

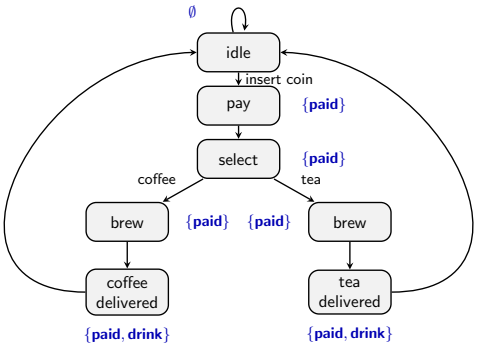
"A drink is never delivered before it is paid for."

"Once it is paid for, a drink is eventually delivered."

"Once a drink is delivered, user has to pay before getting a drink again."

A real transition system: a vending machine

"A vending machine takes a coin, brews coffee or tea, then delivers it."



Atomic propositions $AP = \{\textbf{paid}, \textbf{drink}\}$: the user has paid; a drink is delivered.

Quick questions

1. Which states can follow *select*? two different states, both called *brew*: one for coffee, one for tea!
2. What is the label of *idle*? $\emptyset$: no atomic proposition is true there, a legal label.

Which properties of this machine would we like to state?

"A drink is never delivered before it is paid for."

"Once it is paid for, a drink is eventually delivered."

"Once a drink is delivered, user has to pay before getting a drink again."

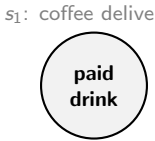
Each one talks about how states follow one another over time: how can we write this in logic?

Topic 23.2

Temporal Connectives

Time by example: next (X)

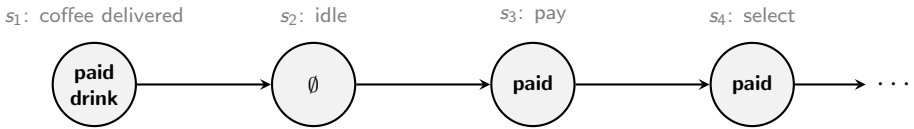
"In a given state, the user has paid (**paid**) and a drink is delivered (**drink**)."



paid $\wedge$ **drink** a formula about *one* state (propositional logic)

Time by example: next (X)

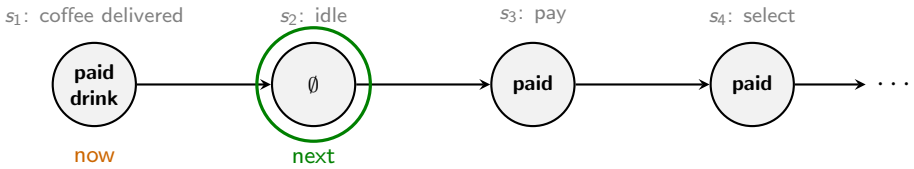
"If a drink is delivered now, the user has to pay again in the *next* state."



drink $\rightarrow$? how do we talk about the *next* state?

Time by example: next (X)

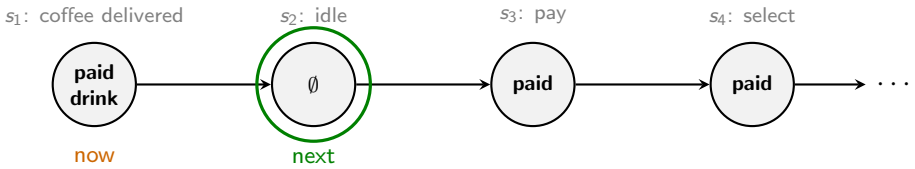
"If a drink is delivered now, the user has to pay again in the *next* state."



drink $\rightarrow X \neg$**paid**

Time by example: next (X)

"If a drink is delivered now, the user has to pay again in the *next* state."

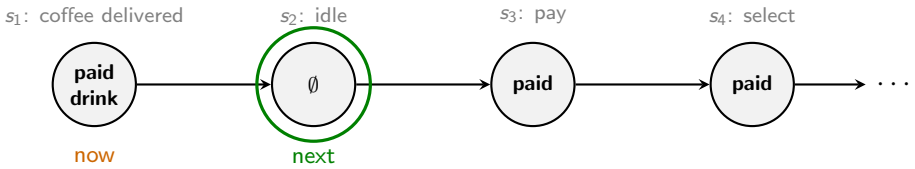


drink $\rightarrow X$ **¬paid**

$X\varphi$: φ holds in the "neXt state"

Time by example: next (X)

"If a drink is delivered now, the user has to pay again in the *next* state."



drink $\rightarrow X \neg$**paid**

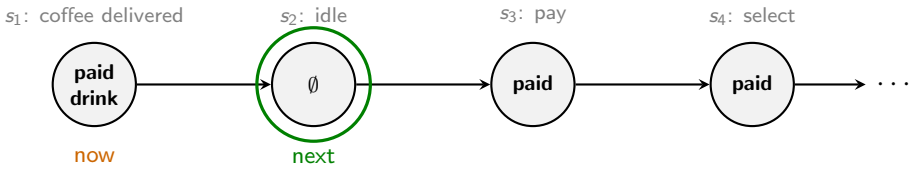
$X\varphi$: φ holds in the "neXt state"

Exercise 23.2 Does the above path, at the current state s_1 (marked *now*), satisfy

1. **drink** $\rightarrow X$ **paid**?
2. **paid** $\rightarrow X$ **paid**?
3. **drink** $\rightarrow X\, X$ **paid**?

Time by example: next (X)

"If a drink is delivered now, the user has to pay again in the *next* state."



drink $\rightarrow X \neg$**paid**

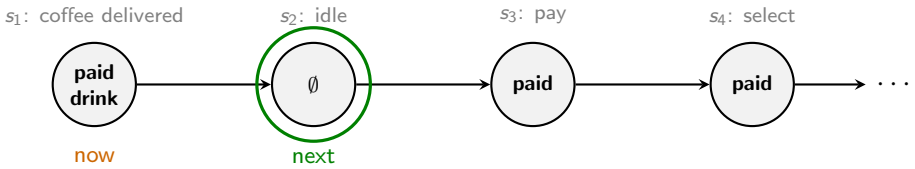
$X\varphi$: φ holds in the "neXt state"

Exercise 23.2 Does the above path, at the current state s_1 (marked *now*), satisfy

1. **drink** $\rightarrow X$ **paid**? No: **paid** does not hold at s_2.
2. **paid** $\rightarrow X$ **paid**?
3. **drink** $\rightarrow X\, X$ **paid**?

Time by example: next (X)

"If a drink is delivered now, the user has to pay again in the *next* state."



drink $\rightarrow X \neg$**paid**

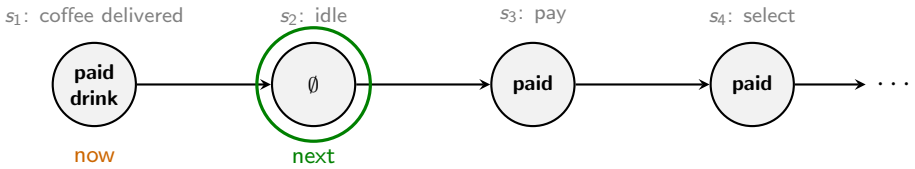
$X\varphi$: φ holds in the "neXt state"

Exercise 23.2 Does the above path, at the current state s_1 (marked *now*), satisfy

1. **drink** $\rightarrow X$ **paid**? No: **paid** does not hold at s_2.
2. **paid** $\rightarrow X$ **paid**? No: **paid** holds at s_1 but not at s_2.
3. **drink** $\rightarrow X\, X$ **paid**?

Time by example: next (X)

"If a drink is delivered now, the user has to pay again in the *next* state."



drink $\rightarrow X \neg$**paid**

$X\varphi$: φ holds in the "neXt state"

Exercise 23.2 Does the above path, at the current state s_1 (marked *now*), satisfy

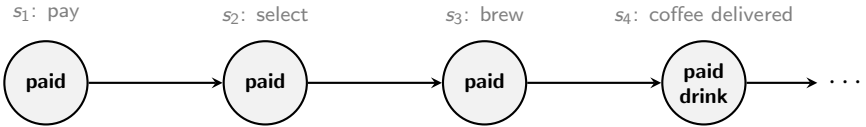
1. **drink** $\rightarrow X$ **paid**? No: **paid** does not hold at s_2.
2. **paid** $\rightarrow X$ **paid**? No: **paid** holds at s_1 but not at s_2.
3. **drink** $\rightarrow X\,X$ **paid**? Yes: **paid** holds at s_3.

Time by example: eventually (F)

"The machine *eventually* delivers a drink (**drink**)."

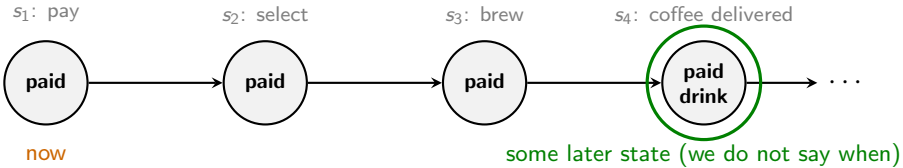
Time by example: eventually (F)

"The machine *eventually* delivers a drink (**drink**)."



Time by example: eventually (F)

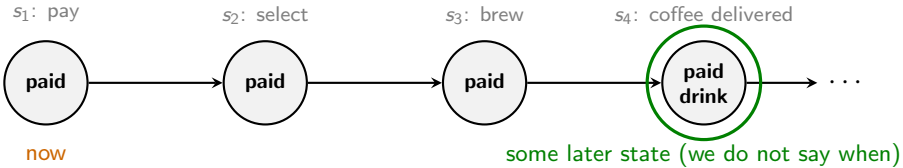
"The machine *eventually* delivers a drink (**drink**)."



F **drink**

Time by example: eventually (F)

"The machine *eventually* delivers a drink (**drink**)."

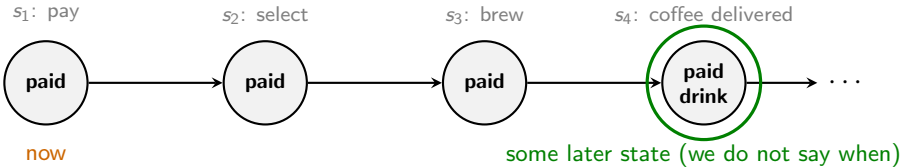


F drink

$F\varphi$: φ holds in "some Future state" (possibly the current one)

Time by example: eventually (F)

"The machine *eventually* delivers a drink (**drink**)."



F **drink**

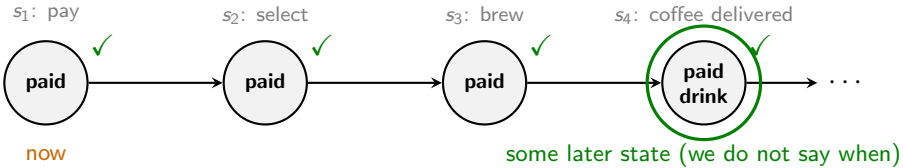
$F\varphi$: φ holds in "some Future state" (possibly the current one)

Exercise 23.3

1. At which states of this path does F **drink** hold?
2. Does F **drink** hold if **drink** already holds in the current state?
3. If **paid** holds now, is a drink eventually delivered on this path?

Time by example: eventually (F)

"The machine *eventually* delivers a drink (**drink**)."



F **drink**

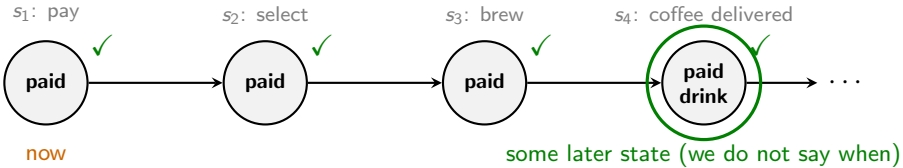
$F\varphi$: φ holds in "some Future state" (possibly the current one)

Exercise 23.3

1. At which states of this path does F **drink** hold? At $s_1, \ldots, s_4$: every state upto and including drink.
2. Does F **drink** hold if **drink** already holds in the current state?
3. If **paid** holds now, is a drink eventually delivered on this path?

Time by example: eventually (F)

"The machine *eventually* delivers a drink (**drink**)."



F **drink**

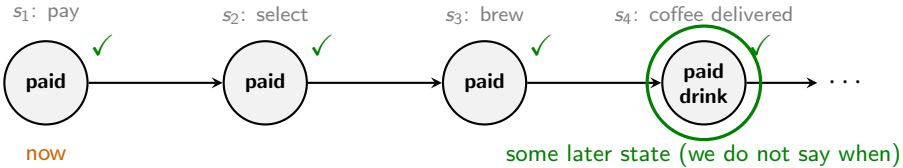
$F\varphi$: φ holds in "some Future state" (possibly the current one)

Exercise 23.3

1. At which states of this path does F **drink** hold? At $s_1, \ldots, s_4$: every state upto and including drink.
2. Does F **drink** hold if **drink** already holds in the current state? Yes: the current state counts.
3. If **paid** holds now, is a drink eventually delivered on this path?

Time by example: eventually (F)

"The machine *eventually* delivers a drink (**drink**)."



F **drink**

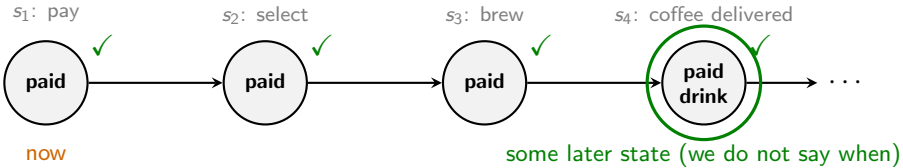
$F\varphi$: φ holds in "some Future state" (possibly the current one)

Exercise 23.3

1. At which states of this path does F **drink** hold? At $s_1, \ldots, s_4$: every state upto and including drink.
2. Does F **drink** hold if **drink** already holds in the current state? Yes: the current state counts.
3. If **paid** holds now, is a drink eventually delivered on this path?
   That is, does **paid** $\rightarrow F$ **drink** hold on this path?

Time by example: eventually (F)

"The machine *eventually* delivers a drink (**drink**)."



F **drink**

$F\varphi$: φ holds in "some Future state" (possibly the current one)

Exercise 23.3

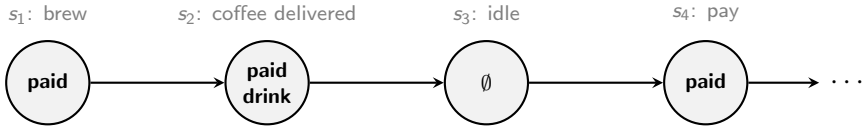
1. At which states of this path does F **drink** hold? At $s_1, \ldots, s_4$: every state upto and including drink.
2. Does F **drink** hold if **drink** already holds in the current state? Yes: the current state counts.
3. If **paid** holds now, is a drink eventually delivered on this path?
   That is, does **paid** $\rightarrow F$ **drink** hold on this path? Yes: **paid** holds at s_1, and so does F **drink**.

Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."

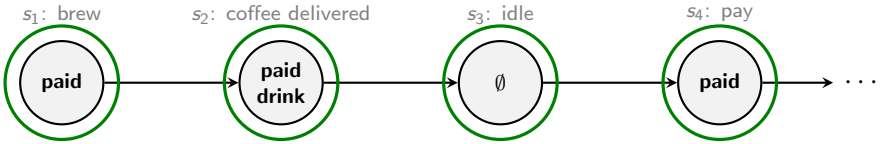
Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."



Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."

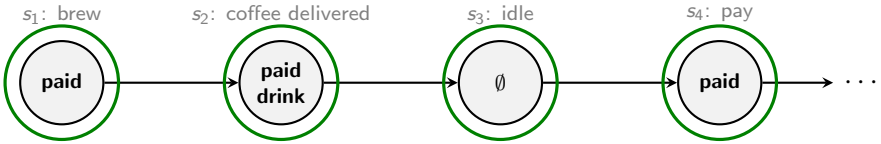


drink $\rightarrow$ **paid** holds in every state, from now on

$$G\,(\textbf{drink} \rightarrow \textbf{paid})$$

Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."



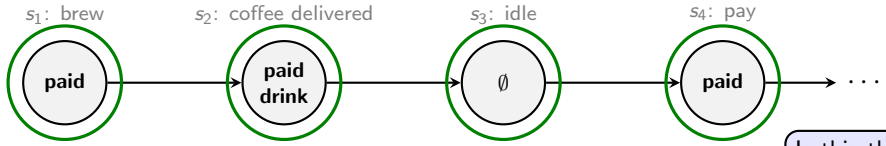
drink $\rightarrow$ **paid** holds in every state, from now on

$$G\,(\textbf{drink} \rightarrow \textbf{paid})$$

$G\varphi$: φ holds "Globally": in the current state and all later ones

Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."



drink $\rightarrow$ **paid** holds in every state, from now on

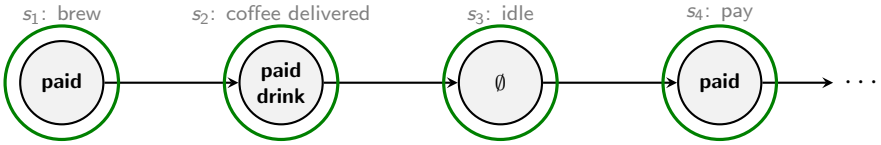
Is this the same as $G($**drink** $\wedge$ **paid**$)$?

$$G\,(\textbf{drink} \rightarrow \textbf{paid})$$

$G\varphi$: φ holds "Globally": in the current state and all later ones

Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."



drink $\rightarrow$ **paid** holds in every state, from now on

$$G\,(\textbf{drink} \rightarrow \textbf{paid})$$

$G\varphi$: φ holds "Globally": in the current state and all later ones

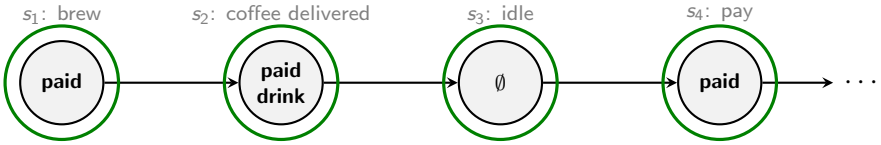
Exercise 23.4

1. Does G **paid** hold on this path?
2. Does G(**paid** $\rightarrow$ **drink**) hold on this path?
3. What would a path look like on which G(**drink** $\rightarrow$ **paid**) fails?

4. Write in LTL: "every time the user has paid, a drink is eventually delivered".

Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."



drink $\rightarrow$ **paid** holds in every state, from now on

$$G\,(\textbf{drink} \rightarrow \textbf{paid})$$

$G\varphi$: φ holds "Globally": in the current state and all later ones

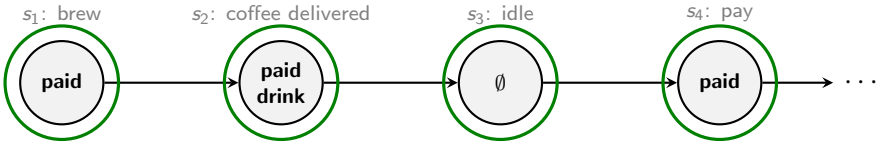
Exercise 23.4

1. Does G **paid** hold on this path? No: **paid** fails at s_3 (idle).
2. Does $G(\textbf{paid} \rightarrow \textbf{drink})$ hold on this path?
3. What would a path look like on which $G(\textbf{drink} \rightarrow \textbf{paid})$ fails?

4. Write in LTL: "every time the user has paid, a drink is eventually delivered".

Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."



drink $\rightarrow$ **paid** holds in every state, from now on

$$G\,(\textbf{drink} \rightarrow \textbf{paid})$$

$G\varphi$: φ holds "Globally": in the current state and all later ones

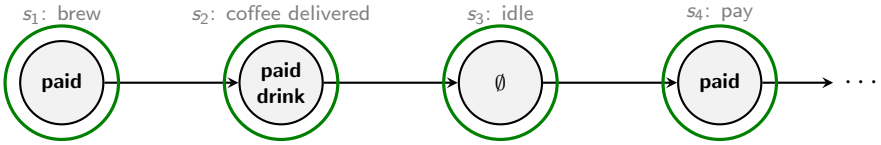
Exercise 23.4

1. Does G **paid** hold on this path? No: **paid** fails at s_3 (idle).
2. Does $G(\textbf{paid} \rightarrow \textbf{drink})$ hold on this path? No: **paid** holds at s_1 but **drink** does not.
3. What would a path look like on which $G(\textbf{drink} \rightarrow \textbf{paid})$ fails?

4. Write in LTL: "every time the user has paid, a drink is eventually delivered".

Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."



drink $\rightarrow$ **paid** holds in every state, from now on

$$G\,(\textbf{drink} \rightarrow \textbf{paid})$$

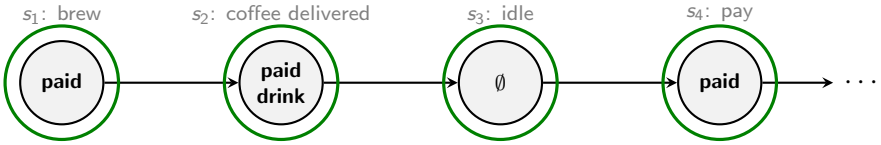
$G\varphi$: φ holds "Globally": in the current state and all later ones

Exercise 23.4

1. Does G **paid** hold on this path? No: **paid** fails at s_3 (idle).
2. Does $G(\textbf{paid} \rightarrow \textbf{drink})$ hold on this path? No: **paid** holds at s_1 but **drink** does not.
3. What would a path look like on which $G(\textbf{drink} \rightarrow \textbf{paid})$ fails? A state where **drink** holds but **paid** does not: a free drink!
4. Write in LTL: "every time the user has paid, a drink is eventually delivered".

Time by example: always (G)

"Whenever a drink is delivered (**drink**), it has been paid for (**paid**)."



drink $\rightarrow$ **paid** holds in every state, from now on

$$G\,(\textbf{drink} \rightarrow \textbf{paid})$$

$G\varphi$: φ holds "Globally": in the current state and all later ones

Exercise 23.4

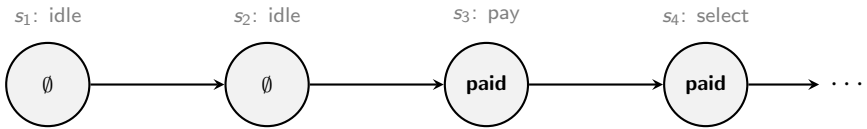
1. Does G **paid** hold on this path? No: **paid** fails at s_3 (idle).
2. Does $G(\textbf{paid} \rightarrow \textbf{drink})$ hold on this path? No: **paid** holds at s_1 but **drink** does not.
3. What would a path look like on which $G(\textbf{drink} \rightarrow \textbf{paid})$ fails? A state where **drink** holds but **paid** does not: a free drink!
4. Write in LTL: "every time the user has paid, a drink is eventually delivered". $G(\textbf{paid} \rightarrow F\,\textbf{drink})$

Time by example: until (U)

"No drink is delivered *until* the user has paid."

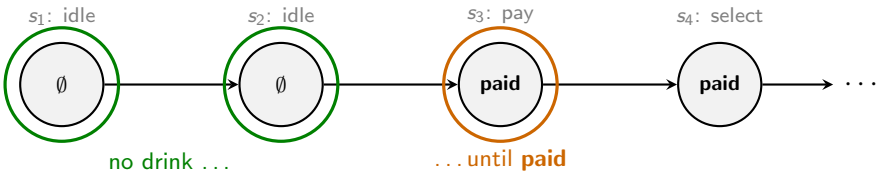
Time by example: until (U)

"No drink is delivered *until* the user has paid."



Time by example: until (U)

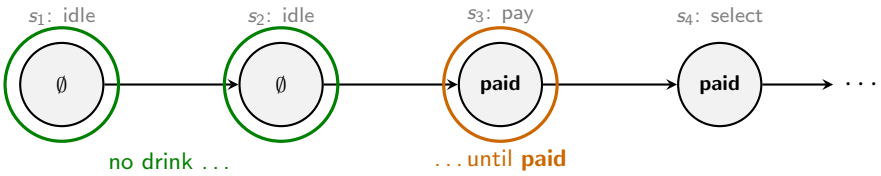
"No drink is delivered *until* the user has paid."



¬drink U paid

Time by example: until (U)

"No drink is delivered *until* the user has paid."

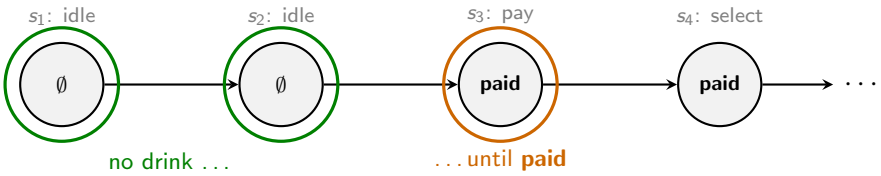


¬drink U paid

$\varphi\, U\, \psi$: φ holds "Until" ψ holds (and ψ does hold)

Time by example: until (U)

"No drink is delivered *until* the user has paid."



¬drink U paid

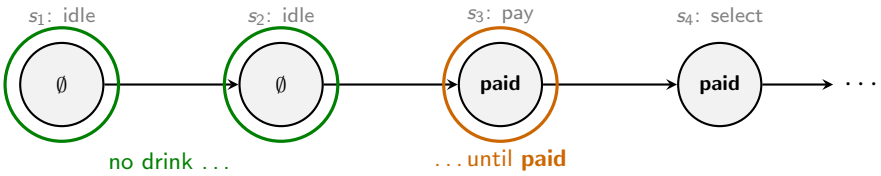
$\varphi\, U\, \psi$: φ holds "Until" ψ holds (and ψ does hold)

Exercise 23.5

1. Does ¬**drink** U **paid** hold at s_2?
2. Does it hold at s_3, where **paid** already holds?
3. If the user never pays (idle forever), does it hold?
4. Which weaker property says "no drink unless the user pays", allowing that the user never pays?

Time by example: until (U)

"No drink is delivered *until* the user has paid."



¬drink U paid

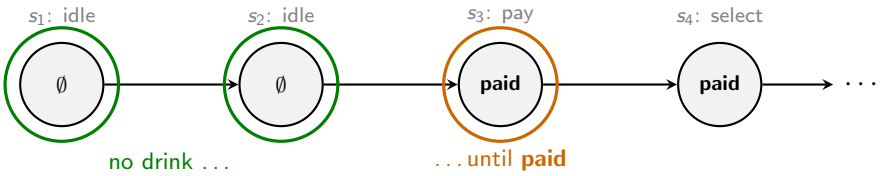
$\varphi\, U\, \psi$: φ holds "Until" ψ holds (and ψ does hold)

Exercise 23.5

1. Does ¬**drink** U **paid** hold at s_2? Yes: **paid** holds at s_3, and ¬**drink** holds at s_2.
2. Does it hold at s_3, where **paid** already holds?
3. If the user never pays (idle forever), does it hold?
4. Which weaker property says "no drink unless the user pays", allowing that the user never pays?

Time by example: until (U)

"No drink is delivered *until* the user has paid."



¬drink U paid

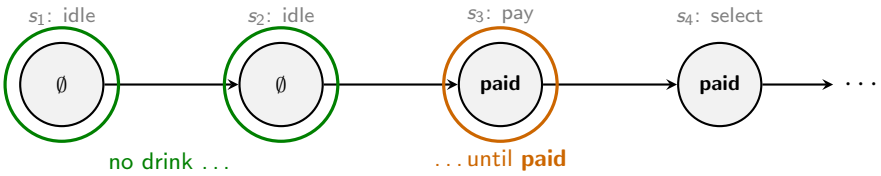
$\varphi\, U\, \psi$: φ holds "Until" ψ holds (and ψ does hold)

Exercise 23.5

1. Does ¬**drink** U **paid** hold at s_2? Yes: **paid** holds at s_3, and ¬**drink** holds at s_2.
2. Does it hold at s_3, where **paid** already holds? Yes: ψ holds immediately, so φ is not needed.
3. If the user never pays (idle forever), does it hold?
4. Which weaker property says "no drink unless the user pays", allowing that the user never pays?

Time by example: until (U)

"No drink is delivered *until* the user has paid."



¬drink U paid

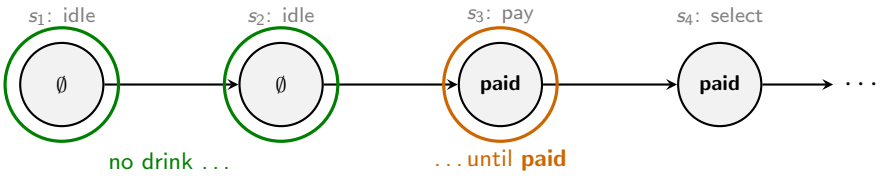
$\varphi\, U\, \psi$: φ holds "Until" ψ holds (and ψ does hold)

Exercise 23.5

1. Does ¬**drink** U **paid** hold at s_2? Yes: **paid** holds at s_3, and ¬**drink** holds at s_2.
2. Does it hold at s_3, where **paid** already holds? Yes: ψ holds immediately, so φ is not needed.
3. If the user never pays (idle forever), does it hold? No: ψ must eventually hold.
4. Which weaker property says "no drink unless the user pays", allowing that the user never pays?

Time by example: until (U)

"No drink is delivered *until* the user has paid."



¬drink U paid

$\varphi\, U\, \psi$: φ holds "Until" ψ holds (and ψ does hold)

Exercise 23.5

1. Does ¬**drink** U **paid** hold at s_2? Yes: **paid** holds at s_3, and ¬**drink** holds at s_2.
2. Does it hold at s_3, where **paid** already holds? Yes: ψ holds immediately, so φ is not needed.
3. If the user never pays (idle forever), does it hold? No: ψ must eventually hold.
4. Which weaker property says "no drink unless the user pays", allowing that the user never pays? ¬**drink** W **paid**, the weak until (as we will see later).

Topic 23.3

LTL Syntax

Linear-time Temporal Logic (LTL): Syntax

The syntax of LTL in Backus–Naur form (BNF):

Linear-time Temporal Logic (LTL): Syntax

The syntax of LTL in Backus–Naur form (BNF):

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi \ U \ \varphi)$$

where p is an atomic proposition from set AP.

Linear-time Temporal Logic (LTL): Syntax

The syntax of LTL in Backus–Naur form (BNF):

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi \ U \ \varphi)$$

where p is an atomic proposition from set AP.

- $\top$, $\bot$ and the usual propositional connectives $\neg, \wedge, \vee, \rightarrow$ are as in propositional logic.
- In addition we have temporal connectives:
  - $X\varphi$: φ holds in the "neXt state"
  - $F\varphi$: φ holds in "some Future state" (possibly the current one)
  - $G\varphi$: φ holds "Globally": in the current state and all later ones
  - $\varphi \, U \, \psi$: φ holds "Until" ψ holds (and ψ does hold)

Linear-time Temporal Logic (LTL): Syntax

The syntax of LTL in Backus–Naur form (BNF):

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi \; U \; \varphi)$$

where p is an atomic proposition from set AP.

- $\top$, $\bot$ and the usual propositional connectives $\neg, \wedge, \vee, \to$ are as in propositional logic.
- In addition we have temporal connectives:
  - $X\varphi$: φ holds in the "neXt state"
  - $F\varphi$: φ holds in "some Future state" (possibly the current one)
  - $G\varphi$: φ holds "Globally": in the current state and all later ones
  - $\varphi \, U \, \psi$: φ holds "Until" ψ holds (and ψ does hold)
- Brackets can be dropped using binding rules ("$>$" reads "binds more tightly than"):

$$\text{Unary } (\neg, X, F, G) \;>\; U \;>\; \vee, \wedge \;>\; \to$$

Parsing LTL formulae

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

Examples

1. $G\ (p \rightarrow F\ q)$

2. $F\ (p \rightarrow G\ r) \vee \neg q\ U\ p$
3. $F\ p \rightarrow G\ r \vee \neg q\ U\ p$

Parsing LTL formulae

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

Examples

1. $G\ (p \to F\ q)$
2. $F\ (p \to G\ r) \vee \neg q\ U\ p$
3. $F\ p \to G\ r \vee \neg q\ U\ p$

Exercise 23.6

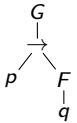
1. Which of these are LTL formulas?
   - 1.1 $(p\ G\ q)$
   - 1.2 $(p\ U\ q)\ U\ r \to\ F\ p$
   - 1.3 $(p \wedge q) \to (U\ r) \vee G\ p$

Parsing LTL formulae

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

Examples

1. $G\ (p \rightarrow F\ q)$



2. $F\ (p \rightarrow G\ r) \vee \neg q\ U\ p$
3. $F\ p \rightarrow G\ r \vee \neg q\ U\ p$

Parse trees

- As usual we can write Parse trees for the formulas

Exercise 23.6

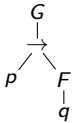
1. Which of these are LTL formulas?
   - 1.1 $(p\ G\ q)$
   - 1.2 $(p\ U\ q)\ U\ r \rightarrow\ F\ p$
   - 1.3 $(p \wedge q) \rightarrow (U\ r) \vee G\ p$

Parsing LTL formulae

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

Examples

1. $G\ (p \rightarrow F\ q)$



2. $F\ (p \rightarrow G\ r) \vee \neg q\ U\ p$
3. $F\ p \rightarrow G\ r \vee \neg q\ U\ p$

Parse trees

- As usual we can write Parse trees for the formulas

Exercise 23.6

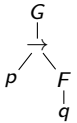
1. Which of these are LTL formulas?
   - 1.1 $(p\ G\ q)$
   - 1.2 $(p\ U\ q)\ U\ r \rightarrow\ F\ p$
   - 1.3 $(p \wedge q) \rightarrow (U\ r) \vee G\ p$
2. Write the parse trees of Examples 2 and 3 and of the formula in part 1 that is an LTL formula.

Parsing LTL formulae

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

Examples

1. $G\ (p \to F\ q)$



2. $F\ (p \to G\ r) \vee \neg q\ U\ p$
3. $F\ p \to G\ r \vee \neg q\ U\ p$

Parse trees

- As usual we can write Parse trees for the formulas
- A subformula of an LTL formula φ is any formula ψ whose parse tree is a subtree of φ's parse tree.

Exercise 23.6

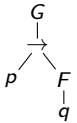
1. Which of these are LTL formulas?
   - 1.1 $(p\ G\ q)$
   - 1.2 $(p\ U\ q)\ U\ r \to\ F\ p$
   - 1.3 $(p \wedge q) \to (U\ r) \vee G\ p$
2. Write the parse trees of Examples 2 and 3 and of the formula in part 1 that is an LTL formula.

Parsing LTL formulae

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

Examples

1. $G\ (p \rightarrow F\ q)$



2. $F\ (p \rightarrow G\ r) \vee \neg q\ U\ p$
3. $F\ p \rightarrow G\ r \vee \neg q\ U\ p$

Parse trees

- As usual we can write Parse trees for the formulas
- A subformula of an LTL formula φ is any formula ψ whose parse tree is a subtree of φ's parse tree.

Exercise 23.6

1. Which of these are LTL formulas?
   - 1.1 $(p\ G\ q)$
   - 1.2 $(p\ U\ q)\ U\ r \rightarrow\ F\ p$
   - 1.3 $(p \wedge q) \rightarrow (U\ r) \vee G\ p$
2. Write the parse trees of Examples 2 and 3 and of the formula in part 1 that is an LTL formula.
3. Write all the subformulas of Example 1 above.

Putting it together: writing specifications

What can you express in this logic?

Consider atomic propositions: **req**, **grant**, **enabled**, **running**.

1. Response: In any state, if there is a **request** (of some resource), then it will be eventually **granted**.

Putting it together: writing specifications

What can you express in this logic?

Consider atomic propositions: **req**, **grant**, **enabled**, **running**.

1. Response: In any state, if there is a **request** (of some resource), then it will be eventually **granted**.
   $G(\textbf{req} \rightarrow F\ \textbf{grant})$

Putting it together: writing specifications

What can you express in this logic?

Consider atomic propositions: **req**, **grant**, **enabled**, **running**.

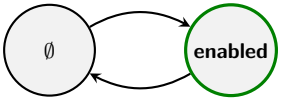
1. Response: In any state, if there is a **request** (of some resource), then it will be eventually **granted**.
   $G(\textbf{req} \rightarrow F\ \textbf{grant})$
2. Some process is **enabled** infinitely often in every computation path.

Putting it together: writing specifications

What can you express in this logic?

Consider atomic propositions: **req**, **grant**, **enabled**, **running**.

1. Response: In any state, if there is a **request** (of some resource), then it will be eventually **granted**.
   $G(\textbf{req} \rightarrow F\ \textbf{grant})$
2. Some process is **enabled** infinitely often in every computation path. $G\ F\ \textbf{enabled}$



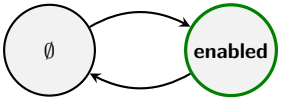
every path keeps coming back to **enabled**

Putting it together: writing specifications

What can you express in this logic?

Consider atomic propositions: **req**, **grant**, **enabled**, **running**.

1. Response: In any state, if there is a **request** (of some resource), then it will be eventually **granted**.
   $G(\textbf{req} \rightarrow F\ \textbf{grant})$
2. Some process is **enabled** infinitely often in every computation path. $G\ F\ \textbf{enabled}$
3. If a process is **enabled** infinitely often, then it **runs** infinitely often. **Exercise 23.7!**



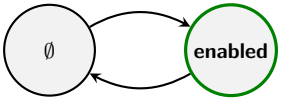
every path keeps coming back to **enabled**

Putting it together: writing specifications

What can you express in this logic?

Consider atomic propositions: **req**, **grant**, **enabled**, **running**.

1. Response: In any state, if there is a **request** (of some resource), then it will be eventually **granted**.
   $G(\textbf{req} \rightarrow F\ \textbf{grant})$
2. Some process is **enabled** infinitely often in every computation path. $G\ F\ \textbf{enabled}$
3. If a process is **enabled** infinitely often, then it **runs** infinitely often. **Exercise 23.7!**



every path keeps coming back to **enabled**

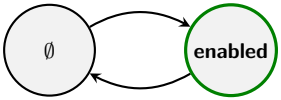
Exercise 23.8 Consider the specification: An upwards moving lift at second floor does not change its direction if someone has pressed button for 5th floor. Come up with atomic propositions and write an LTL formula for this spec.

Putting it together: writing specifications

What can you express in this logic?

Consider atomic propositions: **req**, **grant**, **enabled**, **running**.

1. Response: In any state, if there is a **request** (of some resource), then it will be eventually **granted**.
   $G(\textbf{req} \rightarrow F\ \textbf{grant})$
2. Some process is **enabled** infinitely often in every computation path. $G\ F\ \textbf{enabled}$
3. If a process is **enabled** infinitely often, then it **runs** infinitely often. **Exercise 23.7!**



every path keeps coming back to **enabled**

Exercise 23.8 Consider the specification: An upwards moving lift at second floor does not change its direction if someone has pressed button for 5th floor. Come up with atomic propositions and write an LTL formula for this spec.

safety: nothing bad ever happens, e.g., $G(\textbf{drink} \rightarrow \textbf{paid})$

liveness: something good eventually happens, e.g., $F\ \textbf{drink}$

Topic 23.4

LTL Semantics

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- Semantics are in terms of Transition systems (we also call them models!)
- More precisely, in terms of a path in the model.

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- Semantics are in terms of Transition systems (we also call them models!)
- More precisely, in terms of a path in the model.
- A path in $M = (S, \to, L)$ is an infinite sequence of states $s_1, s_2, \ldots$ in S s.t., $\forall i \geq 1, s_i \to s_{i+1}$. Write $\pi = s_1 \to s_2 \to \ldots$ and denote by π^i the suffix of π starting at s_i.

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi) \mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- A path in $M = (S, \to, L)$ is an infinite sequence of states $s_1, s_2, \ldots$ in S s.t., $\forall i \geq 1, s_i \to s_{i+1}$. Write $\pi = s_1 \to s_2 \to \ldots$ and denote by π^i the suffix of π starting at s_i.

Given LTL formula φ, model $M = (S, \to, L)$ and path $\pi = s_1 \to \ldots$ in M, the satisfaction relation $\pi \models \varphi$, is defined inductively:

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi) \mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- A path in $M = (S, \to, L)$ is an infinite sequence of states $s_1, s_2, \ldots$ in S s.t., $\forall i \geq 1, s_i \to s_{i+1}$. Write $\pi = s_1 \to s_2 \to \ldots$ and denote by π^i the suffix of π starting at s_i.

Given LTL formula φ, model $M = (S, \to, L)$ and path $\pi = s_1 \to \ldots$ in M, the satisfaction relation $\pi \models \varphi$, is defined inductively:

1. $\pi \models \top, \pi \not\models \bot$
2. $\pi \models p$ iff $p \in L(s_1)$
3. $\neg, \wedge, \vee, \to$ are as usual!

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi) \mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- A path in $M = (S, \to, L)$ is an infinite sequence of states $s_1, s_2, \ldots$ in S s.t., $\forall i \geq 1, s_i \to s_{i+1}$. Write $\pi = s_1 \to s_2 \to \ldots$ and denote by π^i the suffix of π starting at s_i.

Given LTL formula φ, model $M = (S, \to, L)$ and path $\pi = s_1 \to \ldots$ in M, the satisfaction relation $\pi \models \varphi$, is defined inductively:

1. $\pi \models \top, \pi \not\models \bot$
2. $\pi \models p$ iff $p \in L(s_1)$
3. $\neg, \wedge, \vee, \to$ are as usual!

4. $\pi \models X\ \varphi$ iff $\pi^2 \models \varphi$.

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi) \mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- A path in $M = (S, \to, L)$ is an infinite sequence of states $s_1, s_2, \ldots$ in S s.t., $\forall i \geq 1, s_i \to s_{i+1}$. Write $\pi = s_1 \to s_2 \to \ldots$ and denote by π^i the suffix of π starting at s_i.

Given LTL formula φ, model $M = (S, \to, L)$ and path $\pi = s_1 \to \ldots$ in M, the satisfaction relation $\pi \models \varphi$, is defined inductively:

1. $\pi \models \top, \pi \not\models \bot$
2. $\pi \models p$ iff $p \in L(s_1)$
3. $\neg, \wedge, \vee, \to$ are as usual!
4. $\pi \models X\ \varphi$ iff $\pi^2 \models \varphi$.
5. $\pi \models F\ \varphi$ iff $\exists i \geq 1$, s.t. $\pi^i \models \varphi$.

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi) \mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- A path in $M = (S, \to, L)$ is an infinite sequence of states $s_1, s_2, \ldots$ in S s.t., $\forall i \geq 1, s_i \to s_{i+1}$. Write $\pi = s_1 \to s_2 \to \ldots$ and denote by π^i the suffix of π starting at s_i.

Given LTL formula φ, model $M = (S, \to, L)$ and path $\pi = s_1 \to \ldots$ in M, the satisfaction relation $\pi \models \varphi$, is defined inductively:

1. $\pi \models \top, \pi \not\models \bot$
2. $\pi \models p$ iff $p \in L(s_1)$
3. $\neg, \wedge, \vee, \to$ are as usual!
4. $\pi \models X\ \varphi$ iff $\pi^2 \models \varphi$.
5. $\pi \models F\ \varphi$ iff $\exists i \geq 1$, s.t. $\pi^i \models \varphi$.
6. $\pi \models G\ \varphi$ iff $\forall i \geq 1$, $\pi^i \models \varphi$.

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi) \mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- A path in $M = (S, \to, L)$ is an infinite sequence of states $s_1, s_2, \ldots$ in S s.t., $\forall i \geq 1, s_i \to s_{i+1}$. Write $\pi = s_1 \to s_2 \to \ldots$ and denote by π^i the suffix of π starting at s_i.

Given LTL formula φ, model $M = (S, \to, L)$ and path $\pi = s_1 \to \ldots$ in M, the satisfaction relation $\pi \models \varphi$, is defined inductively:

1. $\pi \models \top, \pi \not\models \bot$
2. $\pi \models p$ iff $p \in L(s_1)$
3. $\neg, \wedge, \vee, \to$ are as usual!
4. $\pi \models X\ \varphi$ iff $\pi^2 \models \varphi$.
5. $\pi \models F\ \varphi$ iff $\exists i \geq 1$, s.t. $\pi^i \models \varphi$.
6. $\pi \models G\ \varphi$ iff $\forall i \geq 1$, $\pi^i \models \varphi$.
7. $\pi \models \varphi\ U\ \psi$ iff $\exists i \geq 1$, s.t. $\pi^i \models \psi$ and $\forall j \in \{1, \ldots, i-1\}$, $\pi^j \models \varphi$.

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \to \varphi) \mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- A path in $M = (S, \to, L)$ is an infinite sequence of states $s_1, s_2, \ldots$ in S s.t., $\forall i \geq 1, s_i \to s_{i+1}$. Write $\pi = s_1 \to s_2 \to \ldots$ and denote by π^i the suffix of π starting at s_i.

Given LTL formula φ, model $M = (S, \to, L)$ and path $\pi = s_1 \to \ldots$ in M, the satisfaction relation $\pi \models \varphi$, is defined inductively:

1. $\pi \models \top, \pi \not\models \bot$
2. $\pi \models p$ iff $p \in L(s_1)$
3. $\neg, \wedge, \vee, \to$ are as usual!
4. $\pi \models X\ \varphi$ iff $\pi^2 \models \varphi$.
5. $\pi \models F\ \varphi$ iff $\exists i \geq 1$, s.t. $\pi^i \models \varphi$.
6. $\pi \models G\ \varphi$ iff $\forall i \geq 1$, $\pi^i \models \varphi$.
7. $\pi \models \varphi\ U\ \psi$ iff $\exists i \geq 1$, s.t. $\pi^i \models \psi$ and $\forall j \in \{1, \ldots, i-1\}$, $\pi^j \models \varphi$.

Exercise 23.9 If $\pi = s_1 \to s_2 \to s_3 \to \cdots$, what is π^3? Unfold $\pi \models X\,X\,p$ by the definition.

Semantics of LTL

$$\varphi ::= \top \mid \bot \mid p \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$
$$\mid (X\varphi) \mid (F\varphi) \mid (G\varphi) \mid (\varphi U \varphi)$$

- A path in $M = (S, \rightarrow, L)$ is an infinite sequence of states $s_1, s_2, \ldots$ in S s.t., $\forall i \geq 1, s_i \rightarrow s_{i+1}$. Write $\pi = s_1 \rightarrow s_2 \rightarrow \ldots$ and denote by π^i the suffix of π starting at s_i.

Given LTL formula φ, model $M = (S, \rightarrow, L)$ and path $\pi = s_1 \rightarrow \ldots$ in M, the satisfaction relation $\pi \models \varphi$, is defined inductively:

1. $\pi \models \top, \pi \not\models \bot$
2. $\pi \models p$ iff $p \in L(s_1)$
3. $\neg, \wedge, \vee, \rightarrow$ are as usual!
4. $\pi \models X\ \varphi$ iff $\pi^2 \models \varphi$.
5. $\pi \models F\ \varphi$ iff $\exists i \geq 1$, s.t. $\pi^i \models \varphi$.
6. $\pi \models G\ \varphi$ iff $\forall i \geq 1$, $\pi^i \models \varphi$.
7. $\pi \models \varphi\ U\ \psi$ iff $\exists i \geq 1$, s.t. $\pi^i \models \psi$ and $\forall j \in \{1, \ldots, i-1\}$, $\pi^j \models \varphi$.

Exercise 23.9 If $\pi = s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \cdots$, what is π^3? Unfold $\pi \models X\,X\,p$ by the definition. $\pi^3 = s_3 \rightarrow s_4 \rightarrow \cdots$; $\pi \models X\,X\,p$ iff $\pi^2 \models X\,p$ iff $\pi^3 \models p$.

Semantics of LTL continued

- We defined a satisfaction relation between paths and LTL formulas.
- But to verify systems, we would like to say the whole model satisfies a formula.

Semantics of LTL continued

- We defined a satisfaction relation between paths and LTL formulas.
- But to verify systems, we would like to say the whole model satisfies a formula.

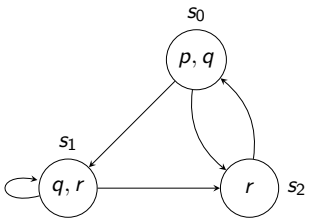
Given LTL formula φ, model $M = (S, \rightarrow, L)$ and $s \in S$, we define $M, s \models \varphi$ if, for every path π of M starting from s, we have $\pi \models \varphi$.

Semantics of LTL continued

- We defined a satisfaction relation between paths and LTL formulas.
- But to verify systems, we would like to say the whole model satisfies a formula.

Given LTL formula φ, model $M = (S, \to, L)$ and $s \in S$, we define $M, s \models \varphi$ if, for every path π of M starting from s, we have $\pi \models \varphi$.

Let's see an example.



Consider M (left). Which of the following hold?

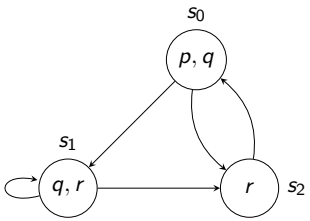
1. $M, s_0 \models (p \wedge q)$
2. $M, s_0 \models \neg r$
3. $M, s_0 \models X\ r$
4. $M, s_0 \models X(q \wedge r)$
5. $M, s_0 \models G\neg(p \wedge r)$

Semantics of LTL continued

- We defined a satisfaction relation between paths and LTL formulas.
- But to verify systems, we would like to say the whole model satisfies a formula.

Given LTL formula φ, model $M = (S, \to, L)$ and $s \in S$, we define $M, s \models \varphi$ if, for every path π of M starting from s, we have $\pi \models \varphi$.

Let's see an example.



Consider M (left). Which of the following hold?

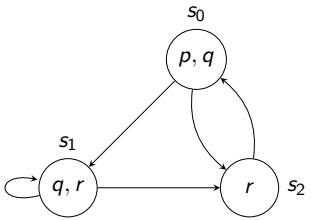
1. $M, s_0 \models (p \wedge q)$ ✓
2. $M, s_0 \models \neg r$ ✓
3. $M, s_0 \models X\ r$
4. $M, s_0 \models X(q \wedge r)$
5. $M, s_0 \models G\neg(p \wedge r)$

Semantics of LTL continued

- We defined a satisfaction relation between paths and LTL formulas.
- But to verify systems, we would like to say the whole model satisfies a formula.

Given LTL formula φ, model $M = (S, \to, L)$ and $s \in S$, we define $M, s \models \varphi$ if, for every path π of M starting from s, we have $\pi \models \varphi$.

Let's see an example.



Consider M (left). Which of the following hold?

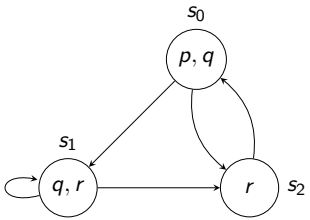
1. $M, s_0 \models (p \wedge q)$ ✓
2. $M, s_0 \models \neg r$ ✓
3. $M, s_0 \models X\ r$
4. $M, s_0 \models X(q \wedge r)$
   ×
5. $M, s_0 \models G\neg(p \wedge r)$

Semantics of LTL continued

- We defined a satisfaction relation between paths and LTL formulas.
- But to verify systems, we would like to say the whole model satisfies a formula.

Given LTL formula φ, model $M = (S, \to, L)$ and $s \in S$, we define $M, s \models \varphi$ if, for every path π of M starting from s, we have $\pi \models \varphi$.

Let's see an example.



Consider M (left).

1. $M, s_0 \models (p \wedge q)$ ✓
2. $M, s_0 \models \neg r$ ✓
3. $M, s_0 \models X\ r$
4. $M, s_0 \models X(q \wedge r)$ ×
5. $M, s_0 \models G\neg(p \wedge r)$

Which of the following hold?

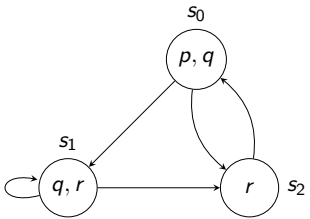
6. $M, s_2 \models G\ r$
7. $\forall s, M, s \models (F(\neg q \wedge r) \to F\ G\ r)$
8. $M, s_0 \models G\ F\ p$
9. $M, s_0 \models (G\ F\ p \to G\ F\ r)$

Semantics of LTL continued

- We defined a satisfaction relation between paths and LTL formulas.
- But to verify systems, we would like to say the whole model satisfies a formula.

Given LTL formula φ, model $M = (S, \to, L)$ and $s \in S$, we define $M, s \models \varphi$ if, for every path π of M starting from s, we have $\pi \models \varphi$.

Let's see an example.



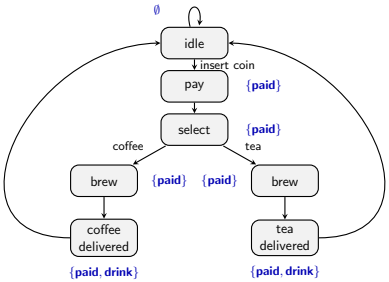
Consider M (left).

1. $M, s_0 \models (p \wedge q)$ ✓
2. $M, s_0 \models \neg r$ ✓
3. $M, s_0 \models X\ r$
4. $M, s_0 \models X(q \wedge r)$ ×
5. $M, s_0 \models G\neg(p \wedge r)$

Which of the following hold?

6. $M, s_2 \models G\ r$
7. $\forall s, M, s \models (F(\neg q \wedge r) \to F\ G\ r)$
8. $M, s_0 \models G\ F\ p$ ×
9. $M, s_0 \models (G\ F\ p \to G\ F\ r)$

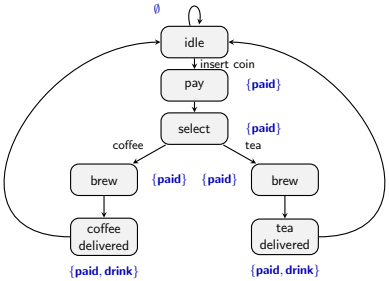
Satisfaction on the vending machine



Exercise 23.10 Which of these hold in the vending machine M (with the labels shown)?

1. $M, idle \models F\,\mathbf{drink}$?
2. $M, pay \models F\,\mathbf{drink}$?
3. $M, idle \models G(\mathbf{drink} \rightarrow \mathbf{paid})$?
4. $M, idle \models (\neg\mathbf{drink}\ U\ \mathbf{paid})$?

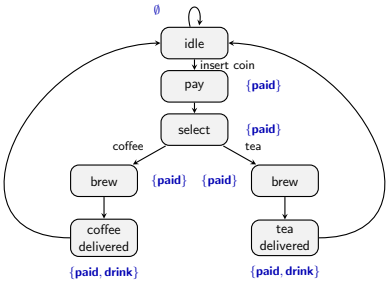
Satisfaction on the vending machine



Exercise 23.10 Which of these hold in the vending machine M (with the labels shown)?

1. $M, idle \models F\,\mathbf{drink}$? No: the path idle, idle, idle, ... never delivers a drink.
2. $M, pay \models F\,\mathbf{drink}$?
3. $M, idle \models G(\mathbf{drink} \rightarrow \mathbf{paid})$?
4. $M, idle \models (\neg\mathbf{drink}\ U\ \mathbf{paid})$?

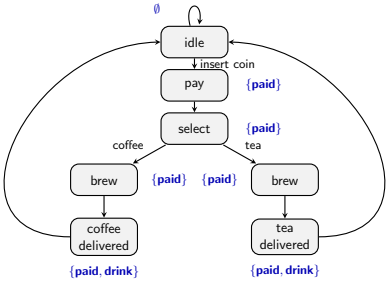
Satisfaction on the vending machine



Exercise 23.10 Which of these hold in the vending machine M (with the labels shown)?

1. $M, idle \models F\,\mathbf{drink}$? No: the path idle, idle, idle, ... never delivers a drink.
2. $M, pay \models F\,\mathbf{drink}$? Yes: every path from pay reaches a delivered state.
3. $M, idle \models G(\mathbf{drink} \to \mathbf{paid})$?
4. $M, idle \models (\neg\mathbf{drink}\ U\ \mathbf{paid})$?

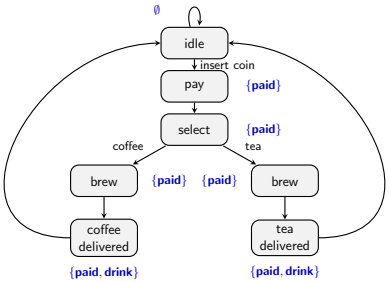
Satisfaction on the vending machine



Exercise 23.10 Which of these hold in the vending machine M (with the labels shown)?

1. $M, idle \models F\,\mathbf{drink}$? No: the path idle, idle, idle, ... never delivers a drink.
2. $M, pay \models F\,\mathbf{drink}$? Yes: every path from pay reaches a delivered state.
3. $M, idle \models G(\mathbf{drink} \to \mathbf{paid})$? Yes: **drink** holds only in the delivered states, and they have **paid**.
4. $M, idle \models (\neg\mathbf{drink}\ U\ \mathbf{paid})$?

Satisfaction on the vending machine



Exercise 23.10 Which of these hold in the vending machine M (with the labels shown)?

1. $M, idle \models F\,\textbf{drink}$? No: the path idle, idle, idle, ... never delivers a drink.
2. $M, pay \models F\,\textbf{drink}$? Yes: every path from pay reaches a delivered state.
3. $M, idle \models G(\textbf{drink} \rightarrow \textbf{paid})$? Yes: **drink** holds only in the delivered states, and they have **paid**.
4. $M, idle \models (\neg\textbf{drink}\ U\ \textbf{paid})$? No: on the path idle, idle, ..., **paid** never holds. (The weak until W would hold.)