Software Validation and Verification
Section II: Model Checking

Topic 4. Linear Temporal Logic

Pedro Cabalar

Department of Computer Science and IT
University of Corunna, SPAIN
cabalar@udc.es

22 de febrero de 2023

Propositional Linear-time Temporal Logic (LTL)

Syntax

- Σ = set of atoms or propositions. Example: $\Sigma = \{p, q, r\}$
- usual propositional operators $\bot, \top, \wedge, \vee, \neg, \rightarrow, \leftrightarrow$
- plus modal operators to talk about (linear) time

Modal operators:

- unary operators:

  $\Box$ = "*forever*"
  $\Diamond$ = "*eventually*"
  $\bigcirc$ = "*next*"

- binary operators:

  $\mathcal{U}$ = "*until*"
  $\mathcal{W}$ = "*until*" (weak version)
  $\mathcal{R}$ = "*release*" (dual of $\mathcal{U}$)

Propositional Linear-time Temporal Logic (LTL)

- Precedence of operators

More priority	$\neg\ \Box\ \Diamond\ \bigcirc$
left assoc.	$\mathcal{U}\ \mathcal{R}\ \mathcal{W}$
	$\wedge$
	$\vee$
	$\rightarrow$
Less priority	$\leftrightarrow$

- Examples:

$$p\ \mathcal{W} \Diamond q \wedge r \quad = \quad (p\ \mathcal{W}\ (\Diamond q)) \wedge r$$

$$\Box p\ \mathcal{U} \neg q\ \mathcal{R}\ r \rightarrow s \quad = \quad \Big(((\Box p)\ \mathcal{U}\ \neg q)\ \mathcal{R}\ r\Big) \rightarrow s$$

Definition 1 (State)

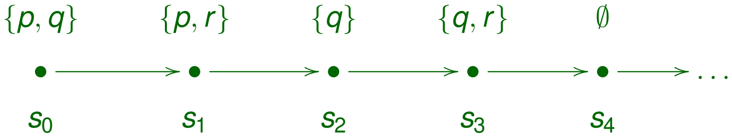
Given a set of propositions Σ, a state s is a truth valuation $s : \Sigma \longrightarrow \{True, False\}$.

It can be represented as the set of (true) atoms. Example: if $\Sigma = \{p, q, r\}$ state $s = \{p, r\}$ means
$s(p) = True, \quad s(q) = False, \quad s(r) = True$.

Definition 2 (Interpretation or trace)

An *interpretation* (or *trace*) M is an infinite sequence of states $s_0, s_1, s_2, \ldots$

Example:



Definition 3 (Satisfaction)

Let $M = s_0, s_1, \ldots$ with $i \geq 0$. We say that $M, i \models \alpha$ when:

- $M, i \models p$ if $p \in s_i$ (for $p \in \Sigma$)
- $M, i \models \Box\alpha$ if $M, j \models \alpha$ for all $j \geq i$
- $M, i \models \Diamond\alpha$ if $M, j \models \alpha$ for some $j \geq i$
- $M, i \models \bigcirc\alpha$ if $M, i+1 \models \alpha$
- $M, i \models \alpha\, \mathcal{U}\, \beta$ if there exists $n \geq i$, $M, n \models \beta$ and $M, j \models \alpha$ for all $i \leq j < n$.
- $M, i \models \alpha\, \mathcal{W}\, \beta$ if $M, i \models \Box\alpha$ or $M, i \models \alpha\, \mathcal{U}\, \beta$

Semantics

- $\bigcirc\alpha$

- $\Box\alpha$

- $\Diamond\alpha$

Semantics

- $\alpha \, \mathcal{U} \, \beta$ = repeat α until (mandatorily) β

- $\alpha \, \mathcal{R} \, \beta$ = there is a α before any state in which $\neg\beta$

Semantics

- $\top \, \mathcal{U} \, \beta$ = repeat $\top$ until (mandatorily) β

  This is equivalent to $\Diamond\beta$.

- $\bot \, \mathcal{R} \, \beta$ = there is a $\bot$ before any state with $\neg\beta$.
  That is, we cannot have $\neg\phi$, i.e., β must hold forever $\Box\beta$

Some standard logical terminology

- Interpretation M is a *model* of theory Γ, written $M \models \Gamma$, iff $M, 0 \models \alpha$ for each formula $\alpha \in \Gamma$.
- Formula α is inconsistent or unsatisfiable iff it has no models. α is a tautology or is valid iff any interpretation is a model of α.
- α is a "logical consequence of" or "is entailed by" Γ, written $\Gamma \models \alpha$, iff any model of Γ satisfies α. Therefore, when $\Gamma = \emptyset$, what does $\models \alpha$ mean?
- Two formulas are equivalent iff they have the same models.
- LTL satisfies $\{\alpha\} \models \beta$ iff $\models \alpha \rightarrow \beta$

  In particular, α and β are equivalent iff $\models \alpha \leftrightarrow \beta$.

Some interesting equivalences

$$\Diamond\alpha \leftrightarrow \top\,\mathcal{U}\,\alpha \quad (1)$$

$$\Box\alpha \leftrightarrow \bot\,\mathcal{R}\,\alpha \quad (2)$$

$$\Box\alpha \leftrightarrow \neg\Diamond\neg\alpha \quad (3)$$

$$\Diamond\alpha \leftrightarrow \neg\Box\neg\alpha \quad (4)$$

$$\Box\alpha \leftrightarrow \alpha \wedge \bigcirc\Box\alpha \quad (5)$$

$$\Diamond\alpha \leftrightarrow \alpha \vee \bigcirc\Diamond\alpha \quad (6)$$

$$\alpha\,\mathcal{U}\,\beta \leftrightarrow (\alpha\,\mathcal{W}\,\beta) \wedge \Diamond\beta \quad (7)$$

$$\alpha\,\mathcal{W}\,\beta \leftrightarrow (\alpha\,\mathcal{U}\,\beta) \vee \Box\alpha \quad (8)$$

$$\alpha\,\mathcal{U}\,\beta \leftrightarrow \beta \vee \alpha \wedge \bigcirc(\alpha\,\mathcal{U}\,\beta) \quad (9)$$

$$\alpha\,\mathcal{R}\,\beta \leftrightarrow \neg(\neg\alpha\,\mathcal{U}\,\neg\beta) \quad (10)$$

$$\alpha\,\mathcal{R}\,\beta \leftrightarrow \beta\,\mathcal{W}\,(\alpha \wedge \beta) \quad (11)$$

(Monadic) First Order Logic

- LTL can be seen as a fragment of First Order Logic (predicate calculus)
- $MFO(<)$ = Monadic First Order Logic with $<$ relation
  - All predicates are monadic (1 argument) $p(x), q(y), \dots$
  - except binary (infix) predicate $x \leq y$, a linear ordering
  - arguments x, y represent time points
  - constant 0 represents initial time point
- Example: $\Box p$ can be translated as $\forall x (x \geq 0 \rightarrow p(x))$

(Monadic) First Order Logic

- We adopt some useful abbreviations

$$
\begin{aligned}
x = y &\stackrel{def}{=} x \leq y \wedge y \leq x \\
x < y &\stackrel{def}{=} x \leq y \wedge \neg(y \leq x) \\
x \leq y \leq z &\stackrel{def}{=} x \leq y \wedge y \leq z \\
\exists x \geq i : \alpha(x) &\stackrel{def}{=} \exists x(i \leq x \wedge \alpha(x)) \\
\forall x \geq i : \alpha(x) &\stackrel{def}{=} \forall x(i \leq x \rightarrow \alpha(x)) \\
\exists x \in i..j : \alpha(x) &\stackrel{def}{=} \exists x(i \leq x \leq j \wedge \alpha(x)) \\
\forall x \in i..j : \alpha(x) &\stackrel{def}{=} \forall x(i \leq x \leq j \rightarrow \alpha(x))
\end{aligned}
$$

- We use function '$+1$' whose meaning can be defined with axiom:

$$(x + 1) = y \stackrel{def}{=} x < y \wedge \neg\exists z(y < z \wedge z < x)$$

Kamp's translation

Temporal formula α at time point i becomes $MFO(<)$ formula $\alpha(i)$

$$
\begin{aligned}
(p)(i) &\stackrel{def}{=} p(i)\\
(\neg\alpha)(i) &\stackrel{def}{=} \neg\alpha(i)\\
(\alpha \vee \beta)(i) &\stackrel{def}{=} \alpha(i) \vee \beta(i)\\
(\alpha \wedge \beta)(i) &\stackrel{def}{=} \alpha(i) \wedge \beta(i)\\
(\bigcirc\alpha)(i) &\stackrel{def}{=} \alpha(i+1)\\
(\Diamond\alpha)(i) &\stackrel{def}{=} \exists j \geq i : \alpha(j)\\
(\Box\alpha)(i) &\stackrel{def}{=} \forall j \geq i : \alpha(j)\\
(\alpha\, \mathcal{U}\, \beta)(i) &\stackrel{def}{=} \exists j \geq i : (\beta(j) \wedge (\forall k \in i..j-1 : \alpha(k)))\\
(\alpha\, \mathcal{R}\, \beta)(i) &\stackrel{def}{=} \forall j \geq i : (\beta(j) \vee (\exists k \in i..j-1 : \alpha(k)))
\end{aligned}
$$

Kamp's translation

The translation is correct:

Theorem 4

$M, i \models \alpha$ if and only if $M \models \alpha(i)$ in $MFO(<)$

but in fact . . .

Theorem 5 (Kamp's theorem, 1968)

LTL is exactly as expressive as $MFO(<)$:

- As we saw, any LTL formula can be naturally written in $MFO(<)$
- The real interest of this theorem is the other direction: any $MFO(<)$ formula can be expressed back in LTL

Kamp's translation

- Example: prove the tautology $\neg(\alpha\, \mathcal{U}\, \beta) \leftrightarrow \neg\alpha\, \mathcal{R}\, \neg\beta$
- Assume any arbitrary time point $i \geq 0$. Then:

$$
\begin{aligned}
(\neg(\alpha\, \mathcal{U}\, \beta))(i) &\leftrightarrow \neg\, (\alpha\, \mathcal{U}\, \beta)(i) \\
&\leftrightarrow \neg\exists j \geq i : (\beta(j) \wedge (\forall k \in i..j-1 : \alpha(k))) \\
&\leftrightarrow \forall j \geq i : \neg(\beta(j) \wedge (\forall k \in i..j-1 : \alpha(k))) \\
&\leftrightarrow \forall j \geq i : (\neg\beta(j) \vee \neg(\forall k \in i..j-1 : \alpha(k))) \\
&\leftrightarrow \forall j \geq i : (\neg\beta(j) \vee (\exists k \in i..j-1 : \neg\alpha(k))) \\
&\leftrightarrow \forall j \geq i : ((\neg\beta)(j) \vee (\exists k \in i..j-1 : (\neg\alpha)(k))) \\
&\leftrightarrow (\neg\alpha\, \mathcal{R}\, \neg\beta)(i)
\end{aligned}
$$

- Example 2: prove the tautology $\Box\alpha \leftrightarrow \alpha \wedge \bigcirc\Box\alpha$

$$
\begin{array}{rcl}
(\Box\alpha)(i) & \leftrightarrow & \forall j \geq i : \alpha(j) \\
 & \leftrightarrow & \alpha(i) \wedge \forall j \geq i+1 : \alpha(j) \\
 & \leftrightarrow & \alpha(i) \wedge (\Box\alpha)(i+1) \\
 & \leftrightarrow & \alpha(i) \wedge (\bigcirc\Box\alpha)(i) \\
 & \leftrightarrow & (\alpha \wedge \bigcirc\Box\alpha)(i)
\end{array}
$$

Exercises

Exercise 1

Prove validity of (6) *and* (9).

Exercise 2

Prove the validity of the following formulas:

$$\begin{array}{rcl} \beta & \rightarrow & \Diamond\beta \\ \beta & \rightarrow & \alpha\,\mathcal{U}\,\beta \\ \alpha\,\mathcal{U}\,\beta & \rightarrow & \Diamond\beta \end{array}$$

Exercises

Exercise 3

Which are the models of $\bot\,\mathcal{U} p$*? Which are the models of* $(\bigcirc p)\,\mathcal{U} \neg p$ *?*

Exercise 4

Define an operator $\mathcal{B}$ *("before") so that* $\alpha\ \mathcal{B}\ \beta$ *means*
for any state in which β *will occur, then some* α *will occur before.*

Exercise 5

Try to express the formula whose models satisfy: p *is true in all even states* $0, 2, 4, \dots$ *and false in odd states.*

Exercise 6

Try to express the formula whose models satisfy: p *is true in all even states* $0, 2, 4, \dots$ *varying* p *freely in odd states.*

Outline

Examples of properties specification

Figure out the meaning of these example formulas:

- $\Box((\neg passport \lor \neg ticket) \to \bigcirc \neg board))$

- $\Box(requested \to \Diamond received)$
- $\Box(received \to \bigcirc processed)$
- $\Box(processed \to \Diamond\Box done)$
- "It can't be that we continually resend a request that is never done." The statement: $\Box requested \land \Box\neg done$ should be inconsistent.
  That is, we should be able to derive $\Box requested \to \Diamond done$.

An example: trains crossing

- Railroad, single rail and a road level-crossing.
- Goal: specifying properties to be satisfied.
- Propositions representing events
  - a = “A train is *a*pproaching"
  - c = “A train is *c*rossing"
  - ℓ = “The ℓight is blinking"
  - b = “The *b*arrier is down"

Safety properties

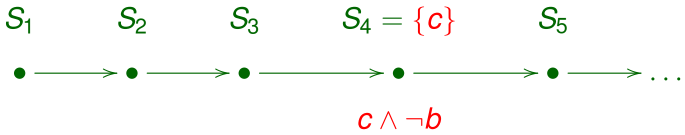
Safety property = something *bad* never happens = $\Box\neg bad$.

- When a train is crossing, the barrier must be down
  Solution: $\Box(c \rightarrow b) \equiv \Box\neg(c \wedge \neg b)$
- If a train is approaching or crossing, the light must be blinking
  Solution: $\Box(a \vee c \rightarrow \ell) \equiv \Box\neg((a \vee c) \wedge \neg\ell)$
- If the barrier is up and the light is off, then no train is coming or crossing. Solution: $\Box(\neg b \wedge \neg\ell \rightarrow \neg a \wedge \neg c) \equiv \Box\neg(\neg b \wedge \neg\ell \wedge (a \vee c))$

Safety properties

Counterexamples of safety properties $\Box \neg bad$

- It suffices with showing finite prefix of the counterexample trace until *bad* occurs
- For instance, a counterexample of $\Box(c \rightarrow b)$ is a trace satisfying $\Diamond(c \wedge \neg b)$



The states from S_5 on are irrelevant and we can only focus on the execution from S_1 to S_4

Liveness properties

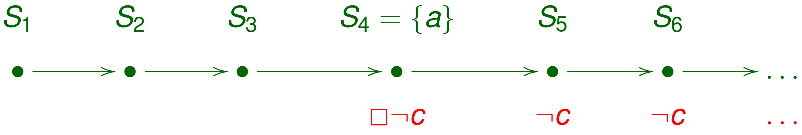
Liveness property = something *initiated* eventually *terminates* = $\Box(initiated \rightarrow \Diamond terminates)$

- When a train is approaching, a train will eventually cross
  Solution: $\Box(a \rightarrow \Diamond c)$
- Sometimes we can use $\mathcal{U}$, $\mathcal{W}$ or $\mathcal{R}$ to propagate a condition until termination.
- When a train is approaching (and nobody is crossing), the barrier will be eventually down before it crosses (if it does so)
  Solution: $\Box(a \wedge \neg c \rightarrow \neg c \, \mathcal{W} \, b)$
- If a train finishes crossing, the barrier will be eventually risen
  Solution: $\Box(c \wedge \bigcirc\neg c \rightarrow \bigcirc\Diamond\neg b)$ Altenative: $\Box\neg(c \wedge c \, \mathcal{U}(\neg c \wedge \Box b))$
  $\equiv \; \Box(c \rightarrow \neg c \, \mathcal{R}(\neg c \rightarrow \Diamond\neg b))$

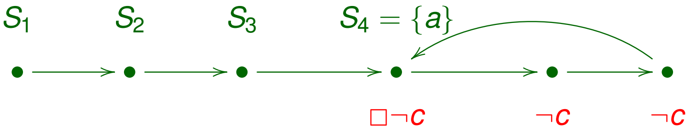
Liveness properties

Counterexamples of liveness properties $\Box(\mathit{initiated} \rightarrow \Diamond \mathit{terminates})$

- A finite prefix does not suffice
- For instance, a counterexample of $\Box(a \rightarrow \Diamond c)$ is a trace satisfying $\Diamond(a \wedge \Box\neg c)$



- Fortunately, in LTL, if a formula has a model (or a countermodel) it also has at least a cyclic model, i.e., it has a periodic prefix that iterates forever



Infinitely often vs latching condition

- Something happens infinitely often = $\Box\Diamond$*something*.
  Example: The barrier is risen infinitely often = $\Box\Diamond\neg b$
- The dual is a latching condition = $\Diamond\Box\alpha$.
  Example: at a given point, no more trains are approaching = $\Diamond\Box\neg a$

Fairness

Fairness means that if a choice holds sufficiently often, then it is taken sufficiently often. Some examples:

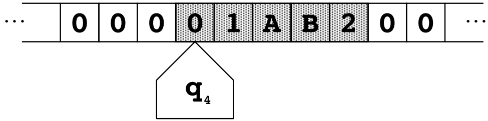
- Unconditional or absolute fairness (a.k.a. impartiality)
  every process should be executed infinitely often $\Box\Diamond executed_i$
- Strong fairness every process enabled infinitely often should be executed infinitely often $\Box\Diamond enabled_i \rightarrow \Box\Diamond executed_i$
- Weak fairness every process permanently enabled after some point should be executed infinitely often
  $\Diamond\Box enabled_i \rightarrow \Box\Diamond executed_i$

Outline

Complexity

- In complexity theory, solving a decision problem means building an algorithm that, in a finite number of steps, answers yes or no to a given input query.
- For instance, *SAT* (propositional satisfiability, i.e., "does a formula α have any model?") is a decision problem, and its complexity class is **NP**-complete.
- Other examples of **NP**-complete problems are: the Travelling Salesman problem, the Graph Coloring problem, Subset Sum problem (find non-empty subset of integers that sum 0).

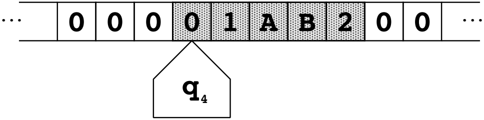
Meaning of NP-completeness



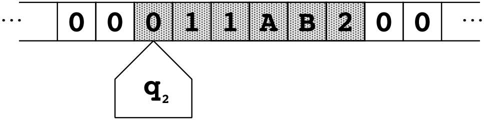
- A Turing Machine (TM) is a theoretical device that operates on an infinite tape with cells containing symbols in a finite alphabet (including the blank or '0')
- The TM has a current state S_i among a finite set of states (including '*Halt*'), and a head pointing to the "current" cell in the tape.
- It has an associated transition function that describes the next step.

Meaning of NP-completeness

- Example: with scanned symbol 0 and state q_4, write 1, move *Left* and go to state q_2. That is:



$$t(0, q_4) = (1, Left, q_2)$$

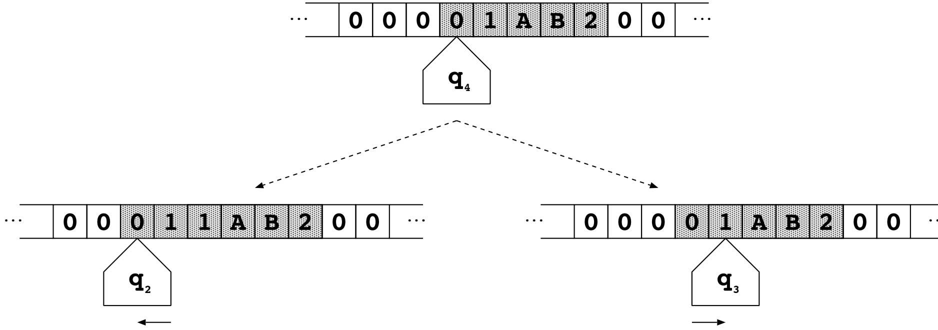


Meaning of NP-completeness

- A decision problem consists in providing a given tape input and asking the Turing Machine for a final output symbol answering *Yes* or *No*.
- Example: *SAT* = given (an encoding of) a propositional formula, does it have at least one model?
- A decision problem is in complexity class **P** iff the number of steps carried out by the TM is polynomial on the size n of the input.

Meaning of NP-completeness

- Now, a non-deterministic Turing Machine (NDTM) is such that the transition function is replaced by a transition relation.
- We may have different possibilities for the next step.
- Example: $t(0, q_4, 1, \mathit{Left}, q_2)$, $t(0, q_4, 0, \mathit{Right}, q_3)$



Meaning of NP-completeness

- Keypoint: an NDTM provides an affirmative answer to a decision problem when at least one of the executions for the same input answers *Yes*.
- A decision problem is in class **NP** iff the number of steps carried out by the NDTM is polynomial on the size n of the input.
- For *SAT*, we can build an NDTM that performs two steps:
  1. For each atom, generate 1 or 0 nondeterministically. This provides an arbitrary interpretation in linear time.
  2. Test whether the current interpretation is a model or not.

  The sequence of these two steps takes polynomial time.

Meaning of NP-completeness

- Unsolved problem

$$\boxed{\mathbf{P} \stackrel{?}{=} \mathbf{NP}}$$

- The most accepted conjecture is that $\mathbf{P} \subset \mathbf{NP}$. But remains unproved.
- It is one of the 7 Millenium Prize Problems
  `http://www.claymath.org/millennium/P_vs_NP/`
  The Clay Mathematics Institute designated $1 million prize for its solution!

Meaning of NP-completeness

- A problem X is **C**-complete, for some complexity class **C**, iff any problem Y in **C** is reducible to X in polynomial-time.
- A complete problem is a representative of the class. Example: if an **NP**-complete problem happened to be in **P** then **P** $=$ **NP**.
- *SAT* was the first problem to be identified as **NP**-complete (Cook's theorem, 1971).
- *SAT* is commonly used nowadays for showing that a problem X is at least as complex as **NP**. To this aim, just encode *SAT* into X.

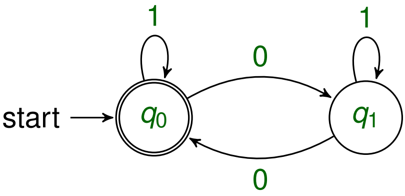
LTL-satisfiability is PSPACE-complete

Theorem 6

[Halpern & Reif 1981], [Sistla & Clarke, 1982]
LTL-satisfiability is **PSPACE***-complete.*

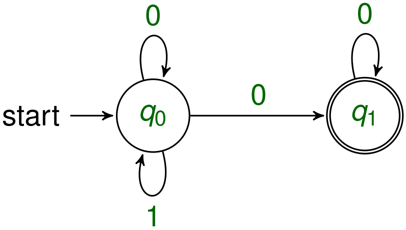
- **PSPACE** is the set of decision problems that can be solved by a Turing Machine using a polynomial amount of space (for a finite, unlimited time).
- There is no difference when the machine is non-deterministic **NPSPACE** $=$ **PSPACE** [Savitch 1970].
- On the other hand, **NP** $\subseteq$ **PSPACE**. Again, unsolved question **NP** $\stackrel{?}{=}$ **PSPACE** but strongly suspected to be $\neq$.
- Other **PSPACE**-complete problems are: Quantified Boolean Formula satisfiability, AI-Planning (STRIPS) existence of plan.

- An finite state machine or finite automaton is a tuple $(Q, A, \delta, q_0, \mathbf{F})$ where
  - Q is a finite set of states
  - A is a finite set called the alphabet
  - $\delta : Q \times A \rightarrow Q$ is the transition function
  - q_0 is the initial state
  - $\mathbf{F}$ is the set of accepting or final states
- Example: this automaton recognizes words containing an even number of 0's



LTL and automata

- ω-automata are a variation where the accepted language consists of words of infinite length. They define different acceptance conditions (when we consider a word to be "accepted")
- A Büchi automaton (BA) is an ω-automaton with the acceptance condition:
  There is some run that visits (at least) one of the states in **F** *infinitely often*
- Example: this automaton recognizes the language $(0+1)^*0^\omega$

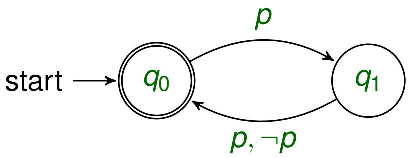


LTL and automata

- During model checking, LTL properties are translated into "equivalent" BA's
- By equivalent we mean they recognize the same language. The BA alphabet A corresponds to the set of possible LTL states.
- Example: if the formula uses atoms $\Sigma = \{p, q\}$ then $A = 2^{\Sigma} = \{\emptyset, \{p\}, \{q\}, \{p, q\}\}$
- Usually, each BA arc is labelled with a set of states that yield the same transition. This set of states is actually represented as an LTL formula.

LTL and automata

- A language accepted by a non-deterministic BA is called regular ω-language.
- An important restriction: LTL is less expressive than Büchi automata.
- For instance, Exercise 6 (make p true in even states and free in all the rest) cannot be represented in *LTL* whereas it is accepted by the Büchi automaton:



- Other temporal logics do cover regular ω-languages.

Outline

Inference Methods

Inference or formal proof: we make syntactic manipulation of formulae. To do so, we use:

- An initial set of formulae: axioms.
- Syntactic manipulation rules: inference rules.
- As a result of applying these rules, we go obtaining new formulae: theorems

Inference methods

- Notation: $\Gamma \vdash \alpha$ means that formula α can be derived or inferred from theory Γ.
- Usually, axioms are not represented inside Γ. Thus, $\vdash \alpha$ means that α is a theorem (from logic **L**).
- Given a language $\mathcal{L}$, a logic **L** is a subset of $\mathcal{L}$. It can be defined:
  - Semantically: $\mathbf{L} = \{\alpha \in \mathcal{L} \mid\ \models \alpha\}$.
  - Syntactically: $\mathbf{L} = \{\alpha \in \mathcal{L} \mid\ \vdash \alpha\}$.
- What should we expect from an inference method?
  - Soundness (or correctness): if $\vdash \alpha$ then $\models \alpha$
  - Completeness: if $\models \alpha$ then $\vdash \alpha$

A deductive system

We define the *LTL* deductive system as follows.

Axioms:

Ax0	*PC*	Any substitution instance of any Propositional Calculus tautology
Ax1	$\vdash \Box(\alpha \to \beta) \to (\Box\alpha \to \Box\beta)$	Distribution of $\Box$ over $\to$
Ax2	$\vdash \bigcirc(\alpha \to \beta) \to (\bigcirc\alpha \to \bigcirc\beta)$	Distribution of $\bigcirc$ over $\to$
Ax3	$\vdash \Box\alpha \to (\alpha \wedge \bigcirc\alpha \wedge \bigcirc\Box\alpha)$	Expansion of $\Box$
Ax4	$\vdash \Box(\alpha \to \bigcirc\alpha) \to (\alpha \to \Box\alpha)$	Induction
Ax5	$\vdash \bigcirc\alpha \leftrightarrow \neg \bigcirc \neg\alpha$	Linearity

Inference rules:

MP	$\frac{\vdash\alpha, \;\; \vdash\alpha\to\beta}{\vdash\beta}$	Modus Ponens
N	$\frac{\vdash\alpha}{\vdash\Box\alpha}$	Necessitation

A deductive system

An example of a proof

Theorem 7 (transitivity)

$\vdash \Box\Box p \leftrightarrow \Box p$

Proof:

1.	$\vdash \Box\Box p \rightarrow \Box p$	Expansion
2.	$\vdash \Box p \rightarrow \bigcirc\Box p$	Expansion
3.	$\vdash \Box(\Box p \rightarrow \bigcirc\Box p)$	Necessitation on 2
4.	$\vdash \Box(\Box p \rightarrow \bigcirc\Box p) \rightarrow (\Box p \rightarrow \Box\Box p)$	Induction
5.	$\vdash \Box p \rightarrow \Box\Box p$	Modus Ponens on $3, 4$
6.	$\vdash \Box\Box p \leftrightarrow \Box p$	P.C. $1, 5$

Q.E.D.

A deductive system

Derived inference rules:

$\mathbf{G}_{\Box}$	$\frac{\vdash \alpha \rightarrow \beta}{\vdash \Box\alpha \rightarrow \Box\beta}$	$\Box$ – Generalization
$\mathbf{G}_{\bigcirc}$	$\frac{\vdash \alpha \rightarrow \beta}{\vdash \bigcirc\alpha \rightarrow \bigcirc\beta}$	$\bigcirc$ – Generalization
Ind	$\frac{\vdash \alpha \rightarrow \bigcirc\alpha}{\vdash \alpha \rightarrow \Box\alpha}$	Induction

These rules can be derived from previous axioms and rules.

A deductive system

Exercises

Exercise 7

Prove the following theorems:

$$\vdash \Box(p \wedge q) \leftrightarrow \Box p \wedge \Box q$$
$$\vdash \Diamond(p \vee q) \leftrightarrow \Diamond p \vee \Diamond q$$

Exercise 8

Prove the theorem

$$\vdash \Box p \vee \Box q \rightarrow \Box(p \vee q)$$

and find a counterexample for:

$$\Box(p \vee q) \rightarrow \Box p \vee \Box q$$

Outline

Semantic tableaux

- For simplicity, we assume $\alpha \to \beta \stackrel{def}{=} \neg\alpha \lor \beta$ and $\alpha \leftrightarrow \beta \stackrel{def}{=} (\alpha \land \beta) \lor (\neg\alpha \land \neg\beta)$
- With respect to Propositional Calculus tableaux, we add unfolding rules for modal operators as follows:

Propositional Calculus rules

Formula	Branch 1	Branch 2
$\alpha \lor \beta$	α	β
$\alpha \land \beta$	α, β	
$\neg(\alpha \lor \beta)$	$\neg\alpha, \neg\beta$	
$\neg(\alpha \land \beta)$	$\neg\alpha$	$\neg\beta$
$\neg\neg\alpha$	α	

Modal rules

Formula	Branch 1	Branch 2
$\Box\alpha$	$\alpha, \bigcirc\Box\alpha$	
$\neg\Diamond\alpha$	$\neg\alpha, \neg\bigcirc\Diamond\alpha$	
$\Diamond\alpha$	α	$\bigcirc\Diamond\alpha$
$\neg\Box\alpha$	$\neg\alpha$	$\neg\bigcirc\Box\alpha$

Semantic tableaux

- When these rules are exhausted, each tableau leaf is boxed and (partially) represents a state
- The state usually contains $\bigcirc$-formulas like $\bigcirc\alpha$ or $\neg\bigcirc\alpha$. In such a case, we generate a transition to a next state whose content is fixed with the new rules:

Formula	Next state
$\bigcirc\alpha$	α
$\neg\bigcirc\alpha$	$\neg\alpha$

- We can reach a state repeated in previous tableau node. If so, we just label the previous node and reuse it

Semantic tableaux

- Example: take $(p \vee q) \wedge \bigcirc(\neg p \wedge \neg q)$

$$(p \vee q) \wedge \bigcirc(\neg p \wedge \neg q)$$

$$(p \vee q), \bigcirc(\neg p \wedge \neg q)$$

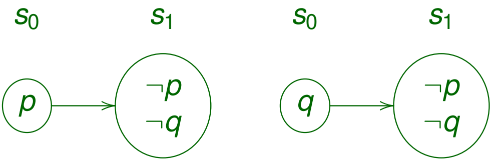
$$\boxed{p, \bigcirc(\neg p \wedge \neg q)} \qquad \boxed{q, \bigcirc(\neg p \wedge \neg q)}$$

- Both open branches yield to a transition to a new state where:

$$\neg p \wedge \neg q$$

$$\boxed{\neg p, \neg q}$$

- That is, any model of $(p \vee q) \wedge \bigcirc(\neg p \wedge \neg q)$ must contain one of the following structures:



- These are called Hintikka structures. They can be expanded to interpretations (arbitrarily completing the truth of the rest of atoms)

Semantic tableaux

- Example 2: is $\Box(p \wedge q) \rightarrow \Box p$ valid?
- We negate the formula and check if we obtain a closed tableau

$$\neg(\Box(p \wedge q) \rightarrow \Box p)$$

$$l_0 : \Box(p \wedge q), \Diamond\neg p$$

$$p \wedge q, \bigcirc\Box(p \wedge q), \Diamond\neg p$$

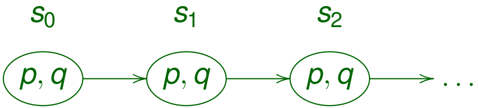
$$p, q, \bigcirc\Box(p \wedge q), \Diamond\neg p$$

$$p, q, \bigcirc\Box(p \wedge q), \neg p \qquad \times$$

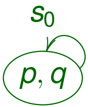
$$p, q, \bigcirc\Box(p \wedge q), \bigcirc\Diamond\neg p$$

We would create a new state with $\Box(p \wedge q), \Diamond\neg p = l_0$

- The tableau is open but generates the following Hintikka structure:



or simply

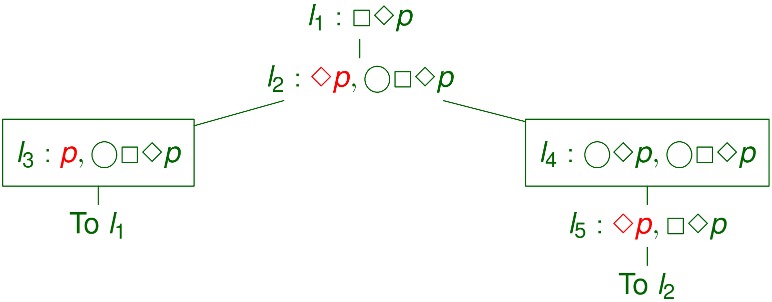


which is never a model because $\Diamond\neg p$ is never fulfilled

- For open tableaux, we will have to check fulfillment of $\Diamond\alpha$ formulas

Semantic tableaux

- Example $\Box\Diamond p$



$\Diamond\alpha$ formulas are fulfilled, so the Hintikka structure represents possible models: