Temporal Logics I: Theory

Daniel Shahaf

Tel-Aviv University

November 2007

Table of Contents

- Next week: Applications.

Motivation for Temporal Logics

- Classical logic is absolute: everything is either true or false.
- And if it is true, it is always true.

Motivation for Temporal Logics

- Classical logic is absolute: everything is either true or false.
- And if it is true, it is always true.
- Life is more complicated.
- Situations change over time.
- Today affects tomorrow.

Motivation for Temporal Logics

- Classical logic is absolute: everything is either true or false.
- And if it is true, it is always true.
- Life is more complicated.
- Situations change over time.
- Today affects tomorrow.
- Need to know what consequences actions today might have tomorrow.
- It is necessary to formalize logic of time-disparate events.
- Such logics are called temporal logics.

What Are Temporal Logics?

- Logics that formalize the notion of “time”.
  - It’s interesting when time is infinite.
- Many variants:

What Are Temporal Logics?

- Logics that formalize the notion of "time".
  - It's interesting when time is infinite.
- Many variants:
  - Branching- or Linear-time.
  - Points or Intervals.
  - Discrete or Continuous.
  - Past or Future.
  - Global or Compositional.
  - Propositional or First-order.
- Uses: concurrent programs verification, circuit modelling, the Elevator Problem. . .

1. Linear-time Temporal Logic
   - Examples
   - Syntax
   - Semantics
   - Comparison to Classical Logic

Linear-time Temporal Logic

- Linear-time temporal logic (LTL) is a discrete-time propositional logic.
- Time has a unique start moment, but no end.

Linear-time Temporal Logic

- Linear-time temporal logic (LTL) is a discrete-time propositional logic.
- Time has a unique start moment, but no end.
- Not perfect:
  - No past-oriented operators.
  - Continuous-time would be better.
- Formally, an instance of modal logic.

Intuition

- LTL is an extension of classical logic.
- It removes nothing, and adds four new connectives:

Intuition

- LTL is an extension of classical logic.
- It removes nothing, and adds four new connectives:
  - $\Box$ Unary, read 'always'. Expresses that something is true henceforth until the end of time.
  - $\Diamond$ Unary, read 'eventually'. Describes things that will definitely happen some day, but does not say when.

Intuition

- LTL is an extension of classical logic.
- It removes nothing, and adds four new connectives:
  - $\bigcirc$ Unary, read 'nexttime'. Talks about what will (or will not) happen at the next point in time.
    In our semantics, a 'next point in time' will always be well-defined.
  - $\mathcal{U}$ Binary, read 'until'. Indicates that one thing will not become false before some other thing becomes true.

Intuition

- LTL is an extension of classical logic.
- It removes nothing, and adds four new connectives:

$\Box$ 'always'	$\bigcirc$ 'nexttime'
$\Diamond$ 'eventually'	$\mathcal{U}$ 'until'

- With these connectives we will be able to discuss issues such as:

Intuition

- LTL is an extension of classical logic.
- It removes nothing, and adds four new connectives:

$\Box$ 'always'	$\bigcirc$ 'nexttime'
$\Diamond$ 'eventually'	$\mathcal{U}$ 'until'

- With these connectives we will be able to discuss issues such as:
  - "The dog ate my homework after I did them."
  - "If you don't eat, a cop will come for you."
  - "Every day it rains in London."
  - "I will continue the diet until I am 70 kg."
  - "I will start studying tomorrow."

The Two Flavours of $\mathcal{U}$

- Two common interpretations to $\mathcal{U}$:
  - Strong until ("$\mathcal{U}_s$"): $\varphi \mathcal{U}_s \psi$ implies $\Diamond\psi$.
  - Weak until ("$\mathcal{U}_w$"): $\Box\varphi$ implies $\varphi \mathcal{U}_w \psi$.
- Both are used.

The Two Flavours of $\mathcal{U}$

- Two common interpretations to $\mathcal{U}$:
  - Strong until ("$\mathcal{U}_s$"): $\varphi \mathcal{U}_s \psi$ implies $\Diamond\psi$.
  - Weak until ("$\mathcal{U}_w$"): $\Box\varphi$ implies $\varphi \mathcal{U}_w \psi$.
- Both are used.
- They are equivalent.
- Each can be expressed in terms of the other:
  - $\varphi \mathcal{U}_s \psi \equiv (\varphi \mathcal{U}_w \psi) \wedge \Diamond\psi$
  - $\varphi \mathcal{U}_w \psi \equiv (\varphi \mathcal{U}_s \psi) \vee \Box\varphi$

Examples

Let $\varphi =$ "Socrates is alive". Suppose that φ holds at $t = 0$.

- "Socrates is mortal":

Examples

Let $\varphi =$ "Socrates is alive". Suppose that φ holds at $t = 0$.

- "Socrates is mortal": $\Diamond\neg\varphi$.
  - Better:

Examples

Let $\varphi =$ "Socrates is alive". Suppose that φ holds at $t = 0$.

- "Socrates is mortal": $\Diamond\neg\varphi$.
  - Better: $\varphi\mathcal{U}_{\mathrm{s}}(\Box\neg\varphi)$.
- "Socrates is immortal":

Examples

Let $\varphi =$ "Socrates is alive". Suppose that φ holds at $t = 0$.

- "Socrates is mortal": $\Diamond\neg\varphi$.
  - Better: $\varphi\mathcal{U}_\mathrm{s}(\Box\neg\varphi)$.
- "Socrates is immortal": $\Box\varphi$.
- "Socrates is immortal":

Examples

Let $\varphi =$ "Socrates is alive". Suppose that φ holds at $t = 0$.

- "Socrates is mortal": $\Diamond\neg\varphi$.
    - Better: $\varphi\mathcal{U}_{\mathrm{s}}(\Box\neg\varphi)$.
- "Socrates is immortal": $\Box\varphi$.
- "Socrates is immortal": $\varphi\mathcal{U}_{\mathrm{w}}\mathbf{F}$.
- "Socrates will be born tomorrow":

Examples

Let $\varphi =$ "Socrates is alive". Suppose that φ holds at $t = 0$.

- "Socrates is mortal": $\Diamond\neg\varphi$.
  - Better: $\varphi\mathcal{U}_{\mathrm{s}}(\Box\neg\varphi)$.
- "Socrates is immortal": $\Box\varphi$.
- "Socrates is immortal": $\varphi\mathcal{U}_{\mathrm{w}}\mathbf{F}$.
- "Socrates will be born tomorrow": $\neg\varphi \wedge \bigcirc\varphi$.

Properties of the Temporal Operators

- Duality: $\Box\neg\varphi \equiv \neg\Diamond\varphi$
- Commutativity: $\bigcirc\Box\varphi \equiv \Box\bigcirc\varphi$ (and likewise for $\Diamond$).
- Distributivity: $\bigcirc(\varphi\mathcal{U}\psi) \equiv (\bigcirc\varphi)\mathcal{U}(\bigcirc\psi)$
- Distributivity: $(p \wedge q)\mathcal{U}_\mathrm{s}\, r \equiv (p\mathcal{U}_\mathrm{s}\, r) \wedge (q\mathcal{U}_\mathrm{s}\, r)$
- Idempotency: $\Box\Box\varphi \equiv \Box\varphi$, $\Diamond\Diamond\varphi \equiv \Diamond\varphi$.
  - The compounds '$\Box\Diamond$' and '$\Diamond\Box$' are idempotent as well.

Properties of the Temporal Operators

- Duality: $\Box\neg\varphi \equiv \neg\Diamond\varphi$
- Commutativity: $\bigcirc\Box\varphi \equiv \Box\bigcirc\varphi$ (and likewise for $\Diamond$).
- Distributivity: $\bigcirc(\varphi\,\mathcal{U}\,\psi) \equiv (\bigcirc\varphi)\,\mathcal{U}\,(\bigcirc\psi)$
- Distributivity: $(p \wedge q)\,\mathcal{U}_\mathrm{s}\, r \equiv (p\,\mathcal{U}_\mathrm{s}\, r) \wedge (q\,\mathcal{U}_\mathrm{s}\, r)$
- Idempotency: $\Box\Box\varphi \equiv \Box\varphi$, $\Diamond\Diamond\varphi \equiv \Diamond\varphi$.
  - The compounds '$\Box\Diamond$' and '$\Diamond\Box$' are idempotent as well.
- Universality of $\mathcal{U}_\mathrm{s}$: $\Diamond\varphi \equiv \mathbf{T}\,\mathcal{U}_\mathrm{s}\,\varphi$.
  $\Longrightarrow$ $\{\neg, \wedge, \bigcirc, \mathcal{U}_\mathrm{s}\}$ is universal.

Properties of the Temporal Operators

- Duality: $\Box\neg\varphi \equiv \neg\Diamond\varphi$
- Commutativity: $\bigcirc\Box\varphi \equiv \Box\bigcirc\varphi$ (and likewise for $\Diamond$).
- Distributivity: $\bigcirc(\varphi\mathcal{U}\psi) \equiv (\bigcirc\varphi)\mathcal{U}(\bigcirc\psi)$
- Distributivity: $(p \wedge q)\mathcal{U}_\mathrm{s}\, r \equiv (p\mathcal{U}_\mathrm{s}\, r) \wedge (q\mathcal{U}_\mathrm{s}\, r)$
- Idempotency: $\Box\Box\varphi \equiv \Box\varphi$, $\Diamond\Diamond\varphi \equiv \Diamond\varphi$.
  - The compounds '$\Box\Diamond$' and '$\Diamond\Box$' are idempotent as well.
- Universality of $\mathcal{U}_\mathrm{s}$: $\Diamond\varphi \equiv \mathbf{T}\mathcal{U}_\mathrm{s}\varphi$.
  $\Longrightarrow \quad \{\neg, \wedge, \bigcirc, \mathcal{U}_\mathrm{s}\}$ is universal.
- Fixpoint characterizations:
  $\Diamond\varphi \equiv \varphi \vee \bigcirc\Diamond\varphi$, $\Box\varphi \equiv \varphi \wedge \bigcirc\Box\varphi$,
  $\varphi\mathcal{U}\psi \equiv \psi \vee (\varphi \wedge \bigcirc(\varphi\mathcal{U}\psi))$.

Properties of the Temporal Operators

- Duality: $\Box\neg\varphi \equiv \neg\Diamond\varphi$
- Commutativity: $\bigcirc\Box\varphi \equiv \Box\bigcirc\varphi$ (and likewise for $\Diamond$).
- Distributivity: $\bigcirc(\varphi\mathcal{U}\psi) \equiv (\bigcirc\varphi)\mathcal{U}(\bigcirc\psi)$
- Distributivity: $(p \wedge q)\mathcal{U}_\mathrm{s} r \equiv (p\mathcal{U}_\mathrm{s} r) \wedge (q\mathcal{U}_\mathrm{s} r)$
- Idempotency: $\Box\Box\varphi \equiv \Box\varphi$, $\Diamond\Diamond\varphi \equiv \Diamond\varphi$.
  - The compounds '$\Box\Diamond$' and '$\Diamond\Box$' are idempotent as well.
- Universality of $\mathcal{U}_\mathrm{s}$: $\Diamond\varphi \equiv \mathbf{T}\mathcal{U}_\mathrm{s}\varphi$.
  $\Longrightarrow$ $\{\neg, \wedge, \bigcirc, \mathcal{U}_\mathrm{s}\}$ is universal.
- Fixpoint characterizations:
  $\Diamond\varphi \equiv \varphi \vee \bigcirc\Diamond\varphi$, $\Box\varphi \equiv \varphi \wedge \bigcirc\Box\varphi$,
  $\varphi\mathcal{U}\psi \equiv \psi \vee (\varphi \wedge \bigcirc(\varphi\mathcal{U}\psi))$.
- $\varphi \wedge \Box(\varphi \rightarrow \bigcirc\varphi) \rightarrow \Box\varphi$

Syntax of LTL

We define a formula recursively as follows:

- Every atomic proposition $p \in P$ is an LTL formula.
- If φ, ψ are LTL formulas, then $\neg\varphi$ and $\varphi \vee \psi$ are LTL formulas.
- If φ, ψ are LTL formulas, then $\Box\varphi$, $\Diamond\varphi$, $\bigcirc\varphi$, and $\varphi\mathfrak{U}\psi$ are LTL formulas.

Syntax of LTL

We define a formula recursively as follows:

- Every atomic proposition $p \in P$ is an LTL formula.
- If φ, ψ are LTL formulas, then $\neg\varphi$ and $\varphi \vee \psi$ are LTL formulas.
- If φ, ψ are LTL formulas, then $\Box\varphi$, $\Diamond\varphi$, $\bigcirc\varphi$, and $\varphi \mathcal{U} \psi$ are LTL formulas.

Examples:

- $p \vee \neg p$ is a formula.
- $p \mathcal{U} \neg \Box q$ is a formula.
- $(\Box(\Diamond\bigcirc p \vee \neg q))\mathcal{U} r$ is a formula.

Syntax of LTL

We define a formula recursively as follows:

- Every atomic proposition $p \in P$ is an LTL formula.
- If φ, ψ are LTL formulas, then $\neg\varphi$ and $\varphi \vee \psi$ are LTL formulas.
- If φ, ψ are LTL formulas, then $\Box\varphi$, $\Diamond\varphi$, $\bigcirc\varphi$, and $\varphi\mathcal{U}\psi$ are LTL formulas.

Examples:

- $p \vee \neg p$ is a formula.
- $p\mathcal{U}\neg\Box q$ is a formula.
- $(\Box(\Diamond\bigcirc p \vee \neg q))\mathcal{U}r$ is a formula.

Alternative notation: $\mathrm{G}\varphi \equiv \Box\varphi$, $\mathrm{F}\varphi \equiv \Diamond\varphi$, $\mathrm{X}\varphi \equiv \bigcirc\varphi$. These stand for "Globally" (or "Generally"), "Future", and "neXt".

Formal Definition of LTL

Let $\mathbb{B} = \{\mathbf{T}, \mathbf{F}\}$.

- A temporal frame is a tuple $\langle S, R \rangle$, where S is a finite or countable non-empty set of states and R is a functional relation imposing a total order on S.
- We assume that every $s \in S$ has an R-successor and denote the latter $R(s)$.
    - Thus, every temporal frame is a frame of modal logic.

Formal Definition of LTL

Let $\mathbb{B} = \{\mathbf{T}, \mathbf{F}\}$.

- A temporal frame is a tuple $\langle S, R \rangle$, where S is a finite or countable non-empty set of states and R is a functional relation imposing a total order on S.
- We assume that every $s \in S$ has an R-successor and denote the latter $R(s)$.
  - Thus, every temporal frame is a frame of modal logic.
- Given a set P of atomic propositions, a temporal interpretation is a tuple $\langle S, R, I \rangle$, where $\langle S, R \rangle$ is a temporal frame and $I \colon S \times P \to \mathbb{B}$ is a temporal interpretation function.
- Given a state (point in time) s and an atomic proposition p, the truth value of p at s is given by $I(s, p)$.

Formal Definition of LTL: Compound Formulas

Let $R^n(s) = R^{n-1}(R(s))$ be the nth successor of s.

The truth value of a compound formula under a temporal interpretation $\mathcal{I} = \langle S, R, I \rangle$ is:

Formal Definition of LTL: Compound Formulas

Let $R^n(s) = R^{n-1}(R(s))$ be the nth successor of s.

The truth value of a compound formula under a temporal interpretation $\mathcal{I} = \langle S, R, I \rangle$ is:

- For classical connectives, nothing changes:
  $I(s, p \wedge q) = I(s, p) \mathbin{\widetilde{\wedge}} I(s, q)$
  $I(s, \neg p) = \widetilde{\neg} I(s, p)$

Formal Definition of LTL: Compound Formulas

Let $R^n(s) = R^{n-1}(R(s))$ be the nth successor of s.

The truth value of a compound formula under a temporal interpretation $\mathcal{I} = \langle S, R, I \rangle$ is:

- For classical connectives, nothing changes:
  $I(s, p \wedge q) = I(s, p) \mathbin{\widetilde{\wedge}} I(s, q)$
  $I(s, \neg p) = \widetilde{\neg} I(s, p)$
- For the universal temporal connectives:
  $I(s, \bigcirc\varphi) = I(R(s), \varphi)$

Formal Definition of LTL: Compound Formulas

Let $R^n(s) = R^{n-1}(R(s))$ be the nth successor of s.

The truth value of a compound formula under a temporal interpretation $\mathcal{I} = \langle S, R, I \rangle$ is:

- For classical connectives, nothing changes:
  $I(s, p \wedge q) = I(s, p) \mathbin{\widetilde{\wedge}} I(s, q)$
  $I(s, \neg p) = \widetilde{\neg} I(s, p)$
- For the universal temporal connectives:
  $I(s, \bigcirc\varphi) = I(R(s), \varphi)$

$$I(s, \varphi \mathcal{U}_{\mathrm{s}} \psi) = \begin{cases} \mathbf{T}, & \text{if } \exists n.\ \Big(I(R^n(s), \psi) \\ & \quad \text{and } \forall 0 \le i < n.\ I(R^i(s), \varphi)\Big); \\ \mathbf{F}, & \text{otherwise.} \end{cases}$$

Formal Definition of LTL: Equivalence to $\mathbb{N}$

Suppose we evaluate $I(s, p)$.

- The only states we can reach from s are $\{R^n(s) \mid n \in \mathbb{N}\}$.
  - We do not and cannot know the past.
- We assumed that S was totally ordered with successors.

Formal Definition of LTL: Equivalence to $\mathbb{N}$

Suppose we evaluate $I(s, p)$.

- The only states we can reach from s are $\{R^n(s) \mid n \in \mathbb{N}\}$.
  - We do not and cannot know the past.
- We assumed that S was totally ordered with successors.
- Therefore, without loss of generality we can assume that $S \simeq \mathbb{N}$.
  - Taking $n \mapsto R^n(s)$.
- Redefine: $I \in \mathbb{N} \times P \to \mathbb{B}$
- Several equivalent views:
  - A subset of $\mathbb{N} \times P$.
  - A sequence of subsets of P.
  - A sequence of classical interpretations.
  - And so on.

Formal Definition of LTL: Examples

- $p \wedge q$ is satisfied by every interpretation that maps both $\langle 0, p \rangle$ and $\langle 0, q \rangle$ to $\mathbf{T}$.
- $\Diamond(\varphi \wedge \bigcirc\psi)$ is satisfied by an interpretation $\mathcal{I}$ iff there is some $n \geq 0$ such that $I(n, \varphi) = I(n+1, \psi) = \mathbf{T}$.
- $(\neg\psi)\mathcal{U}_{\mathrm{w}}\psi$ is valid.
- $(\neg\psi)\mathcal{U}_{\mathrm{s}}\psi$ is not valid.

Past-tense Operators

What happens if we permit past-tense operators on our ray-like timeline?

- The inverses of $\mathcal{U}_{\mathrm{s}}$ and X are sufficient.
  - Their definitions are symmetric.
  - But need to decide how to interpret them at $t = 0$.

Past-tense Operators

What happens if we permit past-tense operators on our ray-like timeline?

- The inverses of $\mathcal{U}_{\mathrm{s}}$ and X are sufficient.
  - Their definitions are symmetric.
  - But need to decide how to interpret them at $t = 0$.
- Increases the language's expressiveness.
  - A formula can "know" what state $\#$ it is evaluated in.
  - Or ask the previous state whether p was true in it.
  - Neither is possible otherwise.

Comparison to Classical Logic

- LTL is a superset of Classical Logic.
- Extends it with the temporal operators.

Comparison to Classical Logic

- LTL is a superset of Classical Logic.
- Extends it with the temporal operators.
- The extension abdicates truth-functionality: Can't tell anything about $\bigcirc\varphi$ from φ itself.
- The same is true for $\mathcal{U}_s$ and friends.

Comparison to Classical Logic

- LTL is a superset of Classical Logic.
- Extends it with the temporal operators.

- The extension abdicates truth-functionality:
  Can't tell anything about $\bigcirc\varphi$ from φ itself.
- The same is true for $\mathcal{U}_s$ and friends.

- The Law of Non-contradiction is more complicated.
  Even if tomorrow is self-contradictory, we might still be sure of today's propositions!

2 Büchi Automata

- ω-Regular Languages
- Büchi Automata
- Properties of Büchi Automata

Infinite Words

- A finite word over an alphabet Σ is a function $w\colon \{0, 1, \ldots, n\} \to \Sigma$.
- Similarly, we may define an infinite (countable) word over Σ as a function $w\colon \mathbb{N} \to \Sigma$.

Infinite Words

- A finite word over an alphabet Σ is a function $w\colon \{0, 1, \ldots, n\} \to \Sigma$.
- Similarly, we may define an infinite (countable) word over Σ as a function $w\colon \mathbb{N} \to \Sigma$.
- We will use the infinite repetition operator to describe infinite words:
- We will write 'p^ω' for the word consisting of $\aleph_0$ repetitions of p.

Infinite Words

- A finite word over an alphabet Σ is a function $w\colon \{0, 1, \dots, n\} \to \Sigma$.
- Similarly, we may define an infinite (countable) word over Σ as a function $w\colon \mathbb{N} \to \Sigma$.
- We will use the infinite repetition operator to describe infinite words:
- We will write 'p^ω' for the word consisting of $\aleph_0$ repetitions of p.

Examples:

- The word '0^ω' is defined by $w(n) = 0$ for all n.
- The word '$(01)^\omega$' is defined by $w(n) = (n \bmod 2)$.
- The word '$14159\dots$' is defined by $w(n-1) =$ the nth decimal digit of π.

ω-Regular Expressions

Reminder:

A regular expression over an alphabet Σ is either:

1. The empty string;
2. An atom $\sigma \in \Sigma$;
3. A concatenation 'pq';
4. An alternation '$p \mid q$';
5. A repetition 'p^*'.

where p, q are (parenthesized) regular expressions.

ω-Regular Expressions

Definition:

An ω-regular expression over an alphabet Σ is either:

1. The empty string;
2. An atom $\sigma \in \Sigma$;
3. A concatenation 'pq';
4. An alternation '$p \mid q$';
5. A finite repetition 'p^*';
6. An infinite repetition 'p^ω'.

where p, q are (parenthesized) ω-regular expressions.

ω-Regular Expressions

Definition:

An ω-regular expression over an alphabet Σ is either:

1. The empty string;
2. An atom $\sigma \in \Sigma$;
3. A concatenation 'pq';
4. An alternation '$p \mid q$' (also written '$p \cup q$');
5. A finite repetition 'p^*';
6. An infinite repetition 'p^ω'.

where p, q are (parenthesized) ω-regular expressions.

Note:

Without loss of generality, we can assume that every ω-regular expression is of the form $\bigcup \alpha_i \beta_i^\omega$, where α_i and β_i are regular regular expressions.

ω-Regular Expressions

Examples:

- Every regular expression is an ω-regular expression.
- $(0 \mid 1)^\omega$ is the set of all infinite words over $\Sigma = \{0, 1\}$.
- 0^ω describes the singleton $\{\lambda n.\, 0\}$.
- $(0 \mid 1)^* \, 1^\omega$ is the set of words that contain only finitely many zeroes.

ω-Regular Expressions

Examples:

- Every regular expression is an ω-regular expression.
- $(0 \mid 1)^\omega$ is the set of all infinite words over $\Sigma = \{0, 1\}$.
- 0^ω describes the singleton $\{\lambda n.\, 0\}$.
- $(0 \mid 1)^* \, 1^\omega$ is the set of words that contain only finitely many zeroes.

Problem cases

- $0^\omega 1$
- $0^\omega 1^\omega$
- $(1^\omega)^*$
- $(1^*)^\omega$
- $(1^\omega)^\omega$

ω-Regular Expressions

Examples:

- Every regular expression is an ω-regular expression.
- $(0 \mid 1)^\omega$ is the set of all infinite words over $\Sigma = \{0, 1\}$.
- 0^ω describes the singleton $\{\lambda n.\, 0\}$.
- $(0 \mid 1)^* \, 1^\omega$ is the set of words that contain only finitely many zeroes.

Problem cases

- $0^\omega 1$
- $0^\omega 1^\omega$
- $(1^\omega)^*$
- $(1^*)^\omega$
- $(1^\omega)^\omega$

Conclusion:

If an ω-regular expression contains an infinite repetition other than at the end, it might be empty, trivial, or undefined.

Büchi Automata

- Büchi automata are a generalization of finite automata to infinite inputs.
  - Proposed by J. R. Büchi in 1962.
- Formally, a Büchi automaton is an NFA.
  - Most concepts—such as 'execution'—carry over unchanged.
- The languages accepted by Büchi automata are a subset of Σ^ω.
  - Note: Σ^ω and Σ^* are disjoint.
  - All words considered are infinite.
- The languages accepted by Büchi automata are called "ω-regular languages".
  - These languages are exactly those accepted by ω-regular expressions.

Büchi Automata: Formal Definition

- A Büchi automaton is a 5-tuple $A = \langle S, \Sigma, \rho, S_0, F \rangle$, where:

$$\begin{array}{rl} S & \text{is a finite set of states;} \\ \Sigma & \text{is an alphabet (finite non-empty set);} \\ \rho & \text{is a transition function;} \\ S_0 & \text{is a set of initial states;} \\ F & \text{is a set of accepting states.} \end{array}$$

- The transition function ρ is $S \times \Sigma \to 2^S$.
- An execution on a word w is a series $s_0, s_1, \ldots$ where $s_0 \in S_0$ and $s_{n+1} \in \rho(s_n, w(n))$ for all n.

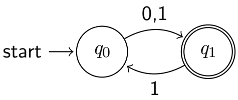
Let $w \in \Sigma^\omega$ and let $s = \{s_i\}_{i \in \mathbb{N}}$ be an execution of A on w.

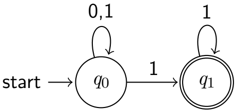
- The execution s accepts the word w iff there is some $f \in F$ such that $s_n = f$ for infinitely many values of n.
- We say that an automaton A accepts a word w if any execution of A on w is accepting.

Let $w \in \Sigma^\omega$ and let $s = \{s_i\}_{i \in \mathbb{N}}$ be an execution of A on w.

- The execution s accepts the word w iff there is some $f \in F$ such that $s_n = f$ for infinitely many values of n.
- We say that an automaton A accepts a word w if any execution of A on w is accepting.

Examples:



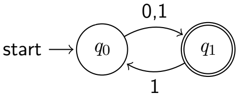


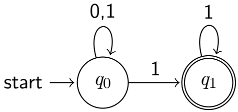
Büchi Automata: Acceptance Criteria

Let $w \in \Sigma^\omega$ and let $s = \{s_i\}_{i \in \mathbb{N}}$ be an execution of A on w.

- The execution s accepts the word w iff there is some $f \in F$ such that $s_n = f$ for infinitely many values of n.
- We say that an automaton A accepts a word w if any execution of A on w is accepting.

Examples:



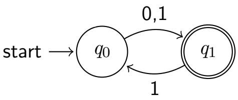


Büchi Automata: Acceptance Criteria

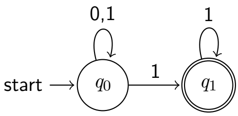
Let $w \in \Sigma^\omega$ and let $s = \{s_i\}_{i \in \mathbb{N}}$ be an execution of A on w.

- The execution s accepts the word w iff there is some $f \in F$ such that $s_n = f$ for infinitely many values of n.
- We say that an automaton A accepts a word w if any execution of A on w is accepting.

Examples:



$L(A) = ((0 \mid 1)\, 1)^\omega$



$L(A) = (0 \mid 1)^*\, 1^\omega$

Generalized Büchi Automata

- We could consider automata of the form $A = \langle S, \Sigma, \rho, S_0, \mathfrak{F} \rangle$ where $\mathfrak{F} = \{F_1, \ldots, F_k\}$ is a set of sets of states.
- An execution would be accepting if it passed infinitely often through every F_i.
- It is sufficient to require that every F_i has some $f_i \in F_i$ that is visited infinitely often.

Generalized Büchi Automata

- We could consider automata of the form $A = \langle S, \Sigma, \rho, S_0, \mathfrak{F} \rangle$ where $\mathfrak{F} = \{F_1, \ldots, F_k\}$ is a set of sets of states.
- An execution would be accepting if it passed infinitely often through every F_i.
- It is sufficient to require that every F_i has some $f_i \in F_i$ that is visited infinitely often.
- Are these more expressive than Büchi automata?

Generalized Büchi Automata

- We could consider automata of the form $A = \langle S, \Sigma, \rho, S_0, \mathcal{F} \rangle$ where $\mathcal{F} = \{F_1, \ldots, F_k\}$ is a set of sets of states.
- An execution would be accepting if it passed infinitely often through every F_i.
- It is sufficient to require that every F_i has some $f_i \in F_i$ that is visited infinitely often.

- Are these more expressive than Büchi automata?
- No; we can construct a Büchi automaton that efficiently simulates a generalized Büchi automaton, as follows:

- Build k copies of the generalized automaton.
- Each copy accepts one F_i.

Properties of Büchi Automata

- Closure under union:
- Closure under intersection:
- Closure under determinization:
- Closure under complementation:

Properties of Büchi Automata

- Closure under union:
  - Trivial:
    $\biguplus \langle S_i, \Sigma, \rho_i, S_{0i}, F_i \rangle = \langle \biguplus S_i, \Sigma, \rho, \biguplus S_{0i}, \biguplus F_i \rangle$
    where $\rho(s_i, \sigma) = \rho_i(s_i, \sigma)$ if $s_i \in S_i$.
- Closure under intersection:
- Closure under determinization:
- Closure under complementation:

Properties of Büchi Automata

- Closure under union:
  - Trivial:
    $\biguplus \langle S_i, \Sigma, \rho_i, S_{0\,i}, F_i \rangle = \langle \biguplus S_i, \Sigma, \rho, \biguplus S_{0\,i}, \biguplus F_i \rangle$
    where $\rho(s_i, \sigma) = \rho_i(s_i, \sigma)$ if $s_i \in S_i$.
- Closure under intersection:
  - Build the cross-product automaton.
  - Not good enough! (Why?)
- Closure under determinization:
- Closure under complementation:

Properties of Büchi Automata

- Closure under union:
  - Trivial:
    $\biguplus \langle S_i, \Sigma, \rho_i, S_{0\,i}, F_i \rangle = \langle \biguplus S_i, \Sigma, \rho, \biguplus S_{0\,i}, \biguplus F_i \rangle$
    where $\rho(s_i, \sigma) = \rho_i(s_i, \sigma)$ if $s_i \in S_i$.
- Closure under intersection:
  - Build the cross-product automaton.
  - Not good enough! (Why?)
  - Solution: build two copies of the cross-product automaton.
- Closure under determinization:
- Closure under complementation:

Properties of Büchi Automata

- Closure under union:
  - Trivial:
    $\biguplus \langle S_i, \Sigma, \rho_i, S_{0i}, F_i \rangle = \langle \biguplus S_i, \Sigma, \rho, \biguplus S_{0i}, \biguplus F_i \rangle$
    where $\rho(s_i, \sigma) = \rho_i(s_i, \sigma)$ if $s_i \in S_i$.
- Closure under intersection:
  - Build the cross-product automaton.
  - Not good enough! (Why?)
  - Solution: build two copies of the cross-product automaton.
- Closure under determinization:
  - Does not hold.
  - Counter-example: $L = (0 \mid 1)^* \, 1^\omega$
  - Proof uses pumping.
- Closure under complementation:

Properties of Büchi Automata

- Closure under union:
  - Trivial:
    $\biguplus \langle S_i, \Sigma, \rho_i, S_{0\,i}, F_i \rangle = \langle \biguplus S_i, \Sigma, \rho, \biguplus S_{0\,i}, \biguplus F_i \rangle$
    where $\rho(s_i, \sigma) = \rho_i(s_i, \sigma)$ if $s_i \in S_i$.
- Closure under intersection:
  - Build the cross-product automaton.
  - Not good enough! (Why?)
  - Solution: build two copies of the cross-product automaton.
- Closure under determinization:
  - Does not hold.
  - Counter-example: $L = (0 \mid 1)^* \, 1^\omega$
  - Proof uses pumping.
- Closure under complementation:
  - Holds—but the complement may be non-deterministic:
  - $L = \left((0 \mid 1)^* \, 0\right)^\omega$

Non-emptiness of Büchi Automata

We wish to decide programmatically whether a given (generalized) Büchi automaton $A = \langle S, \Sigma, \rho, S_0, \mathfrak{F} \rangle$ accepts a non-empty language.

- Let $w \in L(A)$. Consider an accepting run s of A on w.
- Let $f_i \in F_i \in \mathfrak{F}$ be the accepting states through which s passes infinitely.

Non-emptiness of Büchi Automata

We wish to decide programmatically whether a given (generalized) Büchi automaton $A = \langle S, \Sigma, \rho, S_0, \mathfrak{F} \rangle$ accepts a non-empty language.

- Let $w \in L(A)$. Consider an accepting run s of A on w.
- Let $f_i \in F_i \in \mathfrak{F}$ be the accepting states through which s passes infinitely.
- Then all f_i must belong to the same strongly connected component of A.
  - That SCC must be reachable from some starting state $s_0 \in S_0$.
  - And, if $k = 1$, it must contain a cycle through f_1.

Non-emptiness of Büchi Automata

We wish to decide programmatically whether a given (generalized) Büchi automaton $A = \langle S, \Sigma, \rho, S_0, \mathfrak{F} \rangle$ accepts a non-empty language.

- Let $w \in L(A)$. Consider an accepting run s of A on w.
- Let $f_i \in F_i \in \mathfrak{F}$ be the accepting states through which s passes infinitely.
- Then all f_i must belong to the same strongly connected component of A.
  - That SCC must be reachable from some starting state $s_0 \in S_0$.
  - And, if $k = 1$, it must contain a cycle through f_1.
- Clearly, to check non-emptiness, it is sufficient to check that such an SCC exists.
- This may be done in linear time!

3 Automata Recognizing Interpretations

- Overview
- Closures
- Building the Automaton

The Language Defined by a Formula

Recall that an interpretation function $I\colon \mathbb{N} \times P \to \mathbb{B}$ could also be defined as $I\colon \mathbb{N} \to (P \to \mathbb{B})$ or as $I\colon \mathbb{N} \to 2^P$.

- Thus, an interpretation may be viewed as a sequence of subsets of P.
- Or as an infinite word over the alphabet $\Sigma = 2^P$.

The Language Defined by a Formula

Recall that an interpretation function $I\colon \mathbb{N} \times P \to \mathbb{B}$ could also be defined as $I\colon \mathbb{N} \to (P \to \mathbb{B})$ or as $I\colon \mathbb{N} \to 2^P$.

- Thus, an interpretation may be viewed as a sequence of subsets of P.
- Or as an infinite word over the alphabet $\Sigma = 2^P$.
- Büchi automata recognize infinite words over finite alphabets.

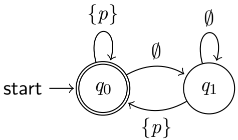
The Language Defined by a Formula

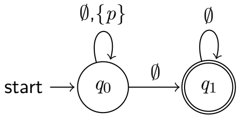
Recall that an interpretation function $I\colon \mathbb{N} \times P \to \mathbb{B}$ could also be defined as $I\colon \mathbb{N} \to (P \to \mathbb{B})$ or as $I\colon \mathbb{N} \to 2^P$.

- Thus, an interpretation may be viewed as a sequence of subsets of P.
- Or as an infinite word over the alphabet $\Sigma = 2^P$.
- Büchi automata recognize infinite words over finite alphabets.
- We will show that Büchi automata can recognize interpretations that satisfy a given LTL formula.
- In other words, we will show that the language defined by a temporal interpretation is an ω-regular language.

Examples:

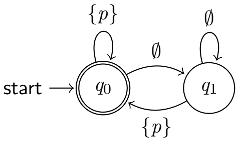
To which formulas do the following automata correspond?



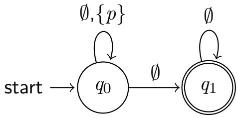


Examples:

To which formulas do the following automata correspond?



$\varphi \equiv \Box\Diamond p$



$\varphi \equiv \Diamond\Box\neg p$

Proof Overview

Goal: To build a Büchi automaton that accepts the set of interpretations $I\colon \mathbb{N} \times P \to \mathbb{B}$ that satisfy a given LTL formula φ.

1. Define closures and closure labellings.
2. Characterise valid closure labellings.
3. Define the automaton in terms of labellings.

Proof Overview

Goal: To build a Büchi automaton that accepts the set of interpretations $I: \mathbb{N} \times P \to \mathbb{B}$ that satisfy a given LTL formula φ.

1. Define closures and closure labellings.
2. Characterise valid closure labellings.
3. Define the automaton in terms of labellings.

Note:

Since emptiness of Büchi automata is decidable, it follows immediately from this construction that satisfiability of LTL formulas is decidable.

Note:

In this proof, $\mathcal{U} \equiv \mathcal{U}_{\mathrm{s}}$.

Detour: More $\mathcal{U}$ Operators

The proof will use the dual operator of $\mathcal{U}_{\mathrm{s}}$, defined by:

$$\varphi \widetilde{\mathcal{U}} \psi \equiv \neg\Big((\neg\varphi)\mathcal{U}_{\mathrm{s}}(\neg\psi)\Big)$$

The $\widetilde{\mathcal{U}}$ operator resembles the weak $\mathcal{U}$ operator:

$$\varphi \widetilde{\mathcal{U}} \psi \equiv \Box\psi \vee \Big(\Diamond(\varphi \wedge \psi) \wedge \psi \mathcal{U}_{\mathrm{s}} \varphi\Big).$$

And has a fixpoint identity:

- $\varphi \widetilde{\mathcal{U}} \psi \equiv \psi \wedge (\varphi \vee \bigcirc(\varphi \widetilde{\mathcal{U}} \psi)).$
- $\varphi \mathcal{U} \psi \equiv \psi \vee (\varphi \wedge \bigcirc(\varphi \mathcal{U} \psi)).$

Definition:

The closure of a temporal formula φ is the smallest set $cl(\varphi)$ such that:

- $\varphi \in cl(\varphi)$
- If $\varphi \in \{\alpha \wedge \beta, \alpha \vee \beta, \bigcirc\alpha, \alpha \mathcal{U} \beta\}$ for some α, β, then $\alpha \in cl(\varphi)$ and $\beta \in cl(\varphi)$.

(The other connectives will be dealt with later.)

Closure Labellings

Definition:

A closure labelling of a sequence $\sigma : \mathbb{N} \to 2^P$ is a mapping $\tau : \mathbb{N} \to 2^{cl(\varphi)}$.

Definition:

A closure labelling of a sequence $\sigma\colon \mathbb{N} \to 2^P$ is a mapping $\tau\colon \mathbb{N} \to 2^{cl(\varphi)}$.

A closure labelling τ of a sequence σ is said to be valid if it satisfies the following conditions for all $i \in \mathbb{N}$:

1. $\mathbf{F} \notin \tau(i)$;

Closure Labellings

Definition:

A closure labelling of a sequence $\sigma\colon \mathbb{N} \to 2^P$ is a mapping $\tau\colon \mathbb{N} \to 2^{cl(\varphi)}$.

A closure labelling τ of a sequence σ is said to be valid if it satisfies the following conditions for all $i \in \mathbb{N}$:

1. $\mathbf{F} \notin \tau(i)$;
2. For each $p \in P$, if $p \in \tau(i)$ then $p \in \sigma(i)$, and if $\neg p \in \tau(i)$ then $p \notin \sigma(i)$;

Closure Labellings

Definition:

A closure labelling of a sequence $\sigma : \mathbb{N} \to 2^P$ is a mapping $\tau : \mathbb{N} \to 2^{cl(\varphi)}$.

A closure labelling τ of a sequence σ is said to be valid if it satisfies the following conditions for all $i \in \mathbb{N}$:

1. $\mathbf{F} \notin \tau(i)$;
2. For each $p \in P$, if $p \in \tau(i)$ then $p \in \sigma(i)$, and if $\neg p \in \tau(i)$ then $p \notin \sigma(i)$;
3. If $\varphi \wedge \psi \in \tau(i)$ then $\varphi \in \sigma(i)$ and $\psi \in \sigma(i)$;
4. If $\varphi \vee \psi \in \tau(i)$ then $\varphi \in \sigma(i)$ or $\psi \in \sigma(i)$;

Closure Labellings

Definition:

A closure labelling of a sequence $\sigma\colon \mathbb{N} \to 2^P$ is a mapping $\tau\colon \mathbb{N} \to 2^{cl(\varphi)}$.

A closure labelling τ of a sequence σ is said to be valid if it satisfies the following conditions for all $i \in \mathbb{N}$:

1. $\mathbf{F} \notin \tau(i)$;
2. For each $p \in P$, if $p \in \tau(i)$ then $p \in \sigma(i)$, and if $\neg p \in \tau(i)$ then $p \notin \sigma(i)$;
3. If $\varphi \wedge \psi \in \tau(i)$ then $\varphi \in \sigma(i)$ and $\psi \in \sigma(i)$;
4. If $\varphi \vee \psi \in \tau(i)$ then $\varphi \in \sigma(i)$ or $\psi \in \sigma(i)$;
5. If $\bigcirc\varphi \in \tau(i)$ then $\varphi \in \tau(i+1)$;

Closure Labellings

Definition:

A closure labelling of a sequence $\sigma\colon \mathbb{N} \to 2^P$ is a mapping $\tau\colon \mathbb{N} \to 2^{cl(\varphi)}$.

A closure labelling τ of a sequence σ is said to be valid if it satisfies the following conditions for all $i \in \mathbb{N}$:

6. If $\varphi \mathcal{U} \psi \in \tau(i)$ then
   either $\psi \in \tau(i)$,
   or $\varphi \in \tau(i)$ and $\varphi \mathcal{U} \psi \in \tau(i+1)$;

Closure Labellings

Definition:

A closure labelling of a sequence $\sigma\colon \mathbb{N} \rightarrow 2^P$ is a mapping $\tau\colon \mathbb{N} \rightarrow 2^{cl(\varphi)}$.

A closure labelling τ of a sequence σ is said to be valid if it satisfies the following conditions for all $i \in \mathbb{N}$:

6. If $\varphi \mathfrak{U} \psi \in \tau(i)$ then
   either $\psi \in \tau(i)$,
   or $\varphi \in \tau(i)$ and $\varphi \mathfrak{U} \psi \in \tau(i+1)$;
7. If $\varphi \widetilde{\mathfrak{U}} \psi \in \tau(i)$
   then $\psi \in \tau(i)$,
   and either $\varphi \in \tau(i)$ or $\varphi \widetilde{\mathfrak{U}} \psi \in \tau(i+1)$;

Closure Labellings

Definition:

A closure labelling of a sequence $\sigma \colon \mathbb{N} \to 2^P$ is a mapping $\tau \colon \mathbb{N} \to 2^{cl(\varphi)}$.

A closure labelling τ of a sequence σ is said to be valid if it satisfies the following conditions for all $i \in \mathbb{N}$:

6. If $\varphi \mathcal{U} \psi \in \tau(i)$ then
   either $\psi \in \tau(i)$,
   or $\varphi \in \tau(i)$ and $\varphi \mathcal{U} \psi \in \tau(i+1)$;
7. If $\varphi \widetilde{\mathcal{U}} \psi \in \tau(i)$
   then $\psi \in \tau(i)$,
   and either $\varphi \in \tau(i)$ or $\varphi \widetilde{\mathcal{U}} \psi \in \tau(i+1)$;
8. If $\varphi \mathcal{U} \psi \in \tau(i)$, then
   $\exists j \geq i$ such that $\psi \in \tau(j)$.

Valid Closure Labellings I

Theorem

If a formula φ is satisfied by a sequence $\sigma : \mathbb{N} \to 2^P$, then there is some valid closure labelling τ such that $\varphi \in \tau(0)$.

Valid Closure Labellings I

Theorem

If a formula φ is satisfied by a sequence $\sigma : \mathbb{N} \to 2^P$, then there is some valid closure labelling τ such that $\varphi \in \tau(0)$.

Proof

Consider the closure labelling given by

$$\tau(n) = \{\psi \in cl(\varphi) \mid \mathcal{I}(n, \psi) = \mathbf{T}\} .$$

Its validity follows immediately from the semantics of LTL. It satisfies $\varphi \in \tau(0)$ since σ satisfies φ.

Valid Closure Labellings II

Theorem

Consider a formula φ *and a sequence* $\sigma : \mathbb{N} \to 2^P$.
If $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ *is a valid closure labelling, then* $\sigma^i \models \bigwedge \tau(i)$.

Valid Closure Labellings II

Theorem

Consider a formula φ and a sequence $\sigma : \mathbb{N} \to 2^P$.
If $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ is a valid closure labelling, then $\sigma^i \models \bigwedge \tau(i)$.

Proof

- For classical connectives and for X it is immediate.

Valid Closure Labellings II

Theorem

Consider a formula φ and a sequence $\sigma : \mathbb{N} \to 2^P$.
If $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ is a valid closure labelling, then $\sigma^i \models \bigwedge \tau(i)$.

Proof

- For classical connectives and for X it is immediate.
- For $\mathcal{U}_\mathrm{s}$: Suppose $\varphi \mathcal{U}_\mathrm{s} \psi \in \tau(i)$.
  Since τ is valid, $\exists j \geq i$ such that $\psi \in \tau(j)$. By the induction hypothesis, $\sigma^j \models \psi$.

Valid Closure Labellings II

Theorem

Consider a formula φ and a sequence $\sigma : \mathbb{N} \to 2^P$.
If $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ is a valid closure labelling, then $\sigma^i \models \bigwedge \tau(i)$.

Proof

- For classical connectives and for X it is immediate.
- For $\mathcal{U}_\mathrm{s}$: Suppose $\varphi \mathcal{U}_\mathrm{s} \psi \in \tau(i)$.
  Since τ is valid, $\exists j \geq i$ such that $\psi \in \tau(j)$. By the induction hypothesis, $\sigma^j \models \psi$.
  Without loss of generality, $\psi \notin \tau(k)$ for every $k \in [i, j)$.
  Again by validity of τ, we obtain that $\varphi \in \tau(k)$ and $\varphi \mathcal{U}_\mathrm{s} \psi \in \tau(k+1)$. Thus, by the inductive hypothesis, $\sigma^{k+1} \models \varphi$ for $i \leq k < j$.

Theorem

Consider a formula φ and a sequence $\sigma : \mathbb{N} \to 2^P$.
If $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ is a valid closure labelling, then $\sigma^i \models \bigwedge \tau(i)$.

Proof

- For classical connectives and for X it is immediate.
- For $\mathcal{U}_\mathrm{s}$: Suppose $\varphi \mathcal{U}_\mathrm{s} \psi \in \tau(i)$.
  Since τ is valid, $\exists j \geq i$ such that $\psi \in \tau(j)$. By the induction hypothesis, $\sigma^j \models \psi$.
  Without loss of generality, $\psi \notin \tau(k)$ for every $k \in [i, j)$.
  Again by validity of τ, we obtain that $\varphi \in \tau(k)$ and $\varphi \mathcal{U}_\mathrm{s} \psi \in \tau(k+1)$. Thus, by the inductive hypothesis, $\sigma^{k+1} \models \varphi$ for $i \leq k < j$.
- For $\widetilde{\mathcal{U}}$, it is immediate, and also follows by duality.

Valid Closure Labellings III

Theorem (Conclusion)

A sequence $\sigma : \mathbb{N} \to 2^P$ *satisfies a formula* φ *if and only if there is valid closure labelling* $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ *of* σ *such that* $\varphi \in \tau(0)$.

Meeting the Requirements: Outline

The theorem specifies two conditions: $\tau(0)$ should contain φ, and $\tau\colon \mathbb{N} \to 2^{cl(\varphi)}$ should be a valid closure labelling of σ.

Meeting the Requirements: Outline

The theorem specifies two conditions: $\tau(0)$ should contain φ, and $\tau\colon \mathbb{N} \to 2^{cl(\varphi)}$ should be a valid closure labelling of σ.

- They give rise to requirements of four kinds: initial conditions; those local to a state; those local to a state and its successor; and eventualities.
- These will be enforced, respectively, by the choice of start states, by the definition of states, by the transition function, and by the accepting states. (The automaton's alphabet is fixed at $\Sigma = 2^P$.)

Meeting the Requirements: Outline

The theorem specifies two conditions: $\tau(0)$ should contain φ, and $\tau\colon \mathbb{N} \to 2^{cl(\varphi)}$ should be a valid closure labelling of σ.

- They give rise to requirements of four kinds: initial conditions; those local to a state; those local to a state and its successor; and eventualities.
- These will be enforced, respectively, by the choice of start states, by the definition of states, by the transition function, and by the accepting states. (The automaton's alphabet is fixed at $\Sigma = 2^P$.)
- The current state of the automaton will correspond to the current label ($\tau(n)$ after n transitions).

Initial conditions

We require that $\varphi \in \tau(0)$. Since $\tau(0)$ is the initial state of an execution of the automaton, we require all initial states to contain φ.

Meeting the Requirements I

Initial conditions

We require that $\varphi \in \tau(0)$. Since $\tau(0)$ is the initial state of an execution of the automaton, we require all initial states to contain φ.

State-local conditions

We require that $\mathbf{F} \notin \tau(i)$ and that if $\varphi \wedge \psi \in \tau(i)$ or $\varphi \vee \psi \in \tau(i)$, then accordingly φ and/or ψ are in $\tau(i)$ as well. Since $\tau(i)$ is a state, we will require all states to have these three properties.

Meeting the Requirements II

Transition conditions

We impose requirements on $\tau(n+1)$ when $\tau(n)$ contains one of $\bigcirc\varphi$, $\varphi\mathfrak{U}_{\mathrm{s}}\psi$, or $\varphi\widetilde{\mathfrak{U}}\psi$.

We will define the transition function in a manner that only allows transitions that meet these requirements.

Further, the transition function also checks that τ corresponds to σ: it validates (by examining the input "letter" $\sigma(i)$) that all atomic propositions in $\tau(i)$ are true and that all false atomic propositions are not in $\tau(i)$.

Eventualities are not checked by the transition function.

Acceptance conditions

The acceptance condition guarantees fulfillment of eventualities.

Since eventualities that are not satisfied at time t reappear at time $t+1$, it is sufficient to check that each eventuality is either satisfied infinitely often or disappears eventually—thus, no "memory" is required.

Thus, for each eventuality $\varphi_i \mathcal{U}_\mathrm{s} \psi_i \in cl(\varphi)$, we require that the execution either passes infinitely through states that contain ($\varphi_i \mathcal{U}_\mathrm{s} \psi_i$ and) ψ_i, or passes infinitely through states that do not contain $\varphi_i \mathcal{U}_\mathrm{s} \psi_i$.

Acceptance conditions

The acceptance condition guarantees fulfillment of eventualities.

Since eventualities that are not satisfied at time t reappear at time $t+1$, it is sufficient to check that each eventuality is either satisfied infinitely often or disappears eventually—thus, no "memory" is required.

Thus, for each eventuality $\varphi_i \mathcal{U}_\mathrm{s} \psi_i \in cl(\varphi)$, we require that the execution either passes infinitely through states that contain ($\varphi_i \mathcal{U}_\mathrm{s} \psi_i$ and) ψ_i, or passes infinitely through states that do not contain $\varphi_i \mathcal{U}_\mathrm{s} \psi_i$.

This is a generalized Büchi condition.

Truth in Advertising

- We have defined an automaton.
- Now, need to prove that it recognizes satisfying interpretations.

Truth in Advertising

- We have defined an automaton.
- Now, need to prove that it recognizes satisfying interpretations.

- It is immediate from the theorem I proved and from the semantics of LTL.
- Filling the details is left as an exercise for the reader.

4 Extensions of Linear-time Temporal Logic

- Motivation
- Defining New Operators
- Restricted Büchi Automata

Motivation

Having shown that Büchi automata are at least as powerful as LTL, let us turn to the converse.

- What LTL formula φ generates $\left\{\sigma \in \mathbb{N} \to 2^P \mid \forall n \in \mathbb{N}.\ p_1 \in \sigma(2n)\right\}$?

Motivation

Having shown that Büchi automata are at least as powerful as LTL, let us turn to the converse.

- What LTL formula φ generates $\left\{\sigma \in \mathbb{N} \to 2^P \mid \forall n \in \mathbb{N}.\ p_1 \in \sigma(2n)\right\}$?
- Problem: How can we tell whether we are in an even state or not?
- How can we assure that φ is true on all odd states—regardless of the values of $\{\sigma(2n+1) \mid n \in \mathbb{N}\}$?

Motivation

Having shown that Büchi automata are at least as powerful as LTL, let us turn to the converse.

- What LTL formula φ generates $\left\{\sigma \in \mathbb{N} \to 2^P \mid \forall n \in \mathbb{N}.\ p_1 \in \sigma(2n)\right\}$?
- Problem: How can we tell whether we are in an even state or not?
- How can we assure that φ is true on all odd states—regardless of the values of $\{\sigma(2n+1) \mid n \in \mathbb{N}\}$?
- We cannot!

Lemma (Wolper '82–'83)

Any temporal logic formula built from an atomic proposition p and containing at most n X-operators has the same truth value for all formulas of the form $p^k(\neg p)p^\omega \quad (k > n)$.

A Closer Look

- Like $\Box$ and $\mathcal{U}_{\mathrm{s}}$, the operator '$\mathrm{even}\,(p)$' may be defined recursively: $\mathrm{even}\,(p) = p \wedge \bigcirc\bigcirc\mathrm{even}\,(p)$
- Convince: it is possible to extend LTL and the construction of Büchi automata from LTL formulas to include this operator—without invalidating the theorem or the axiomatization.

A Closer Look

- Like $\Box$ and $\mathcal{U}_{\mathrm{s}}$, the operator '$\mathrm{even}\,(p)$' may be defined recursively: $\mathrm{even}\,(p) = p \land \bigcirc\bigcirc \mathrm{even}\,(p)$
- Convince: it is possible to extend LTL and the construction of Büchi automata from LTL formulas to include this operator—without invalidating the theorem or the axiomatization.
- The same is true for other definable operators.
  - Divisible by n
  - Divisible by either n or 2
  - True in $t = 0, k, k+\ell, 2k+\ell, 2k+2\ell, 3k+2\ell, \ldots$
  - True iff $t = \alpha k + \beta\ell$ for some α, β (where k, ℓ are given)
- Thus, we want to add all of these to LTL—at the same time.

The Common Description

- All of these may be defined recursively in a similar manner.
- All of these are closed under boolean combinations.
- This is similar to . . . ?

The Common Description

- All of these may be defined recursively in a similar manner.
- All of these are closed under boolean combinations.
- This is similar to linear grammars.
- We will extend LTL by adding operators definable by automata.
    - Most general case: Büchi automata.

The Operator Defined by an Automaton

Let $A = \langle \Sigma, S, \rho, S_0, F \rangle$ be an automaton, where $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ and $n > 0$ is the arity of the operator A.

- Then $A(\varphi_1, \ldots, \varphi_n)$ is true at time t_0 iff there is some $w = \sigma_{w_0}\sigma_{w_1} \ldots \in \Sigma^\omega$ such that φ_{w_j} is true at $t_0 + j$ for every $j \in \mathbb{N}$.

The Operator Defined by an Automaton

Let $A = \langle \Sigma, S, \rho, S_0, F \rangle$ be an automaton, where $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ and $n > 0$ is the arity of the operator A.

- Then $A(\varphi_1, \ldots, \varphi_n)$ is true at time t_0
  iff there is some $w = \sigma_{w_0}\sigma_{w_1} \ldots \in \Sigma^\omega$
  such that φ_{w_j} is true at $t_0 + j$ for every $j \in \mathbb{N}$.

Consider:

- $(\sigma_1\sigma_2 \ldots \sigma_n)^\omega \in A$
- $(\sigma_k)^\omega \in A$
- $A = \mathfrak{U}_\mathrm{s}$, $A = \mathfrak{U}_\mathrm{w}$
- $A = \mathrm{even}$

- "Extended Temporal Logic" (ETL)
  is LTL with automaton-definable operators.

- Both '$\mathcal{U}_{\mathrm{w}}$' and 'even' can be implemented by automata all of whose states are accepting.
- Such automata are looping automata.
  - The subset of ETL they correspond to is known as ETL_{ℓ}.

- Both '$\mathcal{U}_{\mathrm{w}}$' and 'even' can be implemented by automata all of whose states are accepting.
- Such automata are looping automata.
  - The subset of ETL they correspond to is known as ETL_ℓ.
- Theorem (Vardi): ETL_ℓ is equivalent to ETL in terms of expressive power.
- ETL_ℓ is easier to manipulate.
- Complete axiomatizations are known (unlike ETL).

Finite Operators ETL

We now turn to a third variant, ETL_f (for "finite").

- Let $A = \langle \Sigma, S, \rho, S_0, F \rangle$ be an automaton as before.
- Then $A(\varphi_1, \ldots, \varphi_n)$ is true at time t_0 iff there is some $w = \sigma_{w_0}\sigma_{w_1}\ldots\sigma_{w_{k-1}} \in \Sigma^*$ such that φ_{w_j} is true at at $t_0 + j$ for every $j < k$.

Finite Operators ETL

We now turn to a third variant, ETL$_f$ (for "finite").

- Let $A = \langle \Sigma, S, \rho, S_0, F \rangle$ be an automaton as before.
- Then $A(\varphi_1, \ldots, \varphi_n)$ is true at time t_0 iff there is some $w = \sigma_{w_0}\sigma_{w_1} \ldots \sigma_{w_{k-1}} \in \Sigma^*$ such that φ_{w_j} is true at at $t_0 + j$ for every $j < k$.
- ETL$_f$ and ETL$_\ell$ are dual.
- ETL$_f$, ETL$_\ell$, and ETL are equipotent.
- ETL$_f$ has complete axiomatizations.

5 Branching-time Temporal Logic

- Motivation
- Structure
- Formal Definition
- Examples
- Computation Tree Logics

Motivation for BrTL

- Linear time is nice when we are certain about the future.
- However, more often we are uncertain.

Motivation for BrTL

- Linear time is nice when we are certain about the future.
- However, more often we are uncertain.
- We would like to be able to discuss would might happen, not only what definitely will happen.
  - "If it rains tomorrow, will you still come?"
  - "If NASDAQ falls, what will you do?"

Motivation for BrTL

- Linear time is nice when we are certain about the future.
- However, more often we are uncertain.
- We would like to be able to discuss would might happen, not only what definitely will happen.
  - "If it rains tomorrow, will you still come?"
  - "If NASDAQ falls, what will you do?"
- There is more than one possible future.
- Let our logic reflect that.

The Futures

- Informally, the possible futures are represented as a tree.
  - Although less restricted graphs can be considered.

The Futures

- Informally, the possible futures are represented as a tree.
  - Although less restricted graphs can be considered.
- The tree has infinite depth.
- A future is an infinite path on the tree, starting at the root.
  - Formally, an (infinite) sequence of vertices.
- If $\langle u, v \rangle$ is an edge, then v is a possible (immediate) successor to u.
- To uniquely identify a node, we need to know:
  - Some future that contains it.
  - Its index on that future.

Questions to Ask

- It is meaningful to ask:
  - Does some future f_1 of node v satisfy formula φ?
  - Do all futures f_2 of every node on f_1 satisfy φ?
  - Do all nodes on f_1 satisfy ψ?

Questions to Ask

- It is meaningful to ask:
  - Does some future f_1 of node v satisfy formula φ?
  - Do all futures f_2 of every node on f_1 satisfy φ?
  - Do all nodes on f_1 satisfy ψ?
- These questions fall into two fundamentally different categories:
  - Some are concerned with futures of given nodes.
  - Others, with properties of nodes on given futures.
- Accordingly, we will have formulas to describe properties of nodes, and (auxiliary) formulas to describe properties of futures.

Formulas in BrTL

- A state formula is:
  1. An atomic proposition $p \in P$;
  2. A boolean combination of state formulas;
  3. One of $\forall\varphi$, $\exists\varphi$, where φ is a path formula.

Formulas in BrTL

- A state formula is:
  1. An atomic proposition $p \in P$;
  2. A boolean combination of state formulas;
  3. One of $\forall\varphi$, $\exists\varphi$, where φ is a path formula.
- A path formula is:
  1. A state formula;
  2. A boolean combination of path formulas;
  3. One of $\bigcirc\varphi$, $\varphi\mathcal{U}_\mathrm{s}\psi$, where φ and ψ are path formulas.

Formulas in BrTL

- A state formula is:
  1. An atomic proposition $p \in P$;
  2. A boolean combination of state formulas;
  3. One of $\forall\varphi$, $\exists\varphi$, where φ is a path formula.
- A path formula is:
  1. A state formula;
  2. A boolean combination of path formulas;
  3. One of $\bigcirc\varphi$, $\varphi\mathcal{U}_s\psi$, where φ and ψ are path formulas.

- The formulas of BrTL are the state formulas.
- The path formulas are solely an auxiliary.
  Defining them explicitly aids the analysis of state formulas.

Semantics of BrTL

Recall that LTL semantics involve linear temporal interpretations $\langle S, R, I \rangle$.

Semantics of BrTL

Recall that LTL semantics involve linear temporal interpretations $\langle S, R, I\rangle$.

BrTL semantics are a generalization of LTL semantics:

- A (branching-time) temporal frame is a tuple $\langle S, R\rangle$, where S is is a set of states and R is a binary relation on S, such that every $s \in S$ has at least one R-successor.
  - The set of R-successors of s will be written $R(s)$.
- A (branching-time) temporal interpretation function is as before.

Semantics of BrTL

Let φ, ψ be state formulas; α, β be path formulas; p, q be atoms; $s \in S$ be a state; and $\ell \colon \mathbb{N} \to S$ be a path.

- An interpretation $\mathcal{I} = \langle S, R, I \rangle$ assigns a truth-value to every path and state formula.
- For state formulas:

Semantics of BrTL

Let φ, ψ be state formulas; α, β be path formulas; p, q be atoms; $s \in S$ be a state; and $\ell \colon \mathbb{N} \to S$ be a path.

- An interpretation $\mathcal{I} = \langle S, R, I \rangle$ assigns a truth-value to every path and state formula.
- For state formulas:
  - $\mathcal{I}(s, p) = I(s, p)$;

Semantics of BrTL

Let φ, ψ be state formulas; α, β be path formulas; p, q be atoms; $s \in S$ be a state; and $\ell\colon \mathbb{N} \to S$ be a path.

- An interpretation $\mathcal{I} = \langle S, R, I \rangle$ assigns a truth-value to every path and state formula.
- For state formulas:
  - $\mathcal{I}(s, p) = I(s, p)$;
  - $\mathcal{I}(s, \varphi \circ \psi) = \mathcal{I}(s, \varphi) \,\tilde{\circ}\, \mathcal{I}(s, \psi)$;

Semantics of BrTL

Let φ, ψ be state formulas; α, β be path formulas; p, q be atoms; $s \in S$ be a state; and $\ell\colon \mathbb{N} \to S$ be a path.

- An interpretation $\mathcal{I} = \langle S, R, I \rangle$ assigns a truth-value to every path and state formula.
- For state formulas:
  - $\mathcal{I}(s, p) = I(s, p)$;
  - $\mathcal{I}(s, \varphi \circ \psi) = \mathcal{I}(s, \varphi) \,\tilde{\circ}\, \mathcal{I}(s, \psi)$;
  - $\mathcal{I}(s, \forall\alpha) = \mathbf{T}$ iff $\mathcal{I}(\ell, \alpha) = \mathbf{T}$ whenever $\ell(0) = s$;
  - $\mathcal{I}(s, \exists\alpha) = \mathbf{T}$ iff there is some path ℓ starting at s such that $\mathcal{I}(\ell, \alpha) = \mathbf{T}$.

Semantics of BrTL

Let φ, ψ be state formulas; α, β be path formulas; p, q be atoms; $s \in S$ be a state; and $\ell\colon \mathbb{N} \to S$ be a path.

- An interpretation $\mathcal{I} = \langle S, R, I \rangle$ assigns a truth-value to every path and state formula.
- For state formulas:
  - $\mathcal{I}(s, p) = I(s, p)$;
  - $\mathcal{I}(s, \varphi \circ \psi) = \mathcal{I}(s, \varphi) \,\tilde{\circ}\, \mathcal{I}(s, \psi)$;
  - $\mathcal{I}(s, \forall\alpha) = \mathbf{T}$ iff $\mathcal{I}(\ell, \alpha) = \mathbf{T}$ whenever $\ell(0) = s$;
  - $\mathcal{I}(s, \exists\alpha) = \mathbf{T}$ iff there is some path ℓ starting at s such that $\mathcal{I}(\ell, \alpha) = \mathbf{T}$.
- For path formulas:
  - $\mathcal{I}(\ell, \varphi) = \mathcal{I}\big(\ell(0), \varphi\big)$;

Semantics of BrTL

Let φ, ψ be state formulas; α, β be path formulas; p, q be atoms; $s \in S$ be a state; and $\ell \colon \mathbb{N} \to S$ be a path.

- An interpretation $\mathcal{I} = \langle S, R, I \rangle$ assigns a truth-value to every path and state formula.
- For state formulas:
  - $\mathcal{I}(s, p) = I(s, p)$;
  - $\mathcal{I}(s, \varphi \circ \psi) = \mathcal{I}(s, \varphi) \,\tilde{\circ}\, \mathcal{I}(s, \psi)$;
  - $\mathcal{I}(s, \forall\alpha) = \mathbf{T}$ iff $\mathcal{I}(\ell, \alpha) = \mathbf{T}$ whenever $\ell(0) = s$;
  - $\mathcal{I}(s, \exists\alpha) = \mathbf{T}$ iff there is some path ℓ starting at s such that $\mathcal{I}(\ell, \alpha) = \mathbf{T}$.
- For path formulas:
  - $\mathcal{I}(\ell, \varphi) = \mathcal{I}\big(\ell(0), \varphi\big)$;
  - $\mathcal{I}(\ell, \alpha \circ \beta) = \mathcal{I}(\ell, \alpha) \,\tilde{\circ}\, \mathcal{I}(\ell, \beta)$;

Semantics of BrTL

Let φ, ψ be state formulas; α, β be path formulas; p, q be atoms; $s \in S$ be a state; and $\ell\colon \mathbb{N} \to S$ be a path.

- An interpretation $\mathcal{I} = \langle S, R, I \rangle$ assigns a truth-value to every path and state formula.
- For state formulas:
  - $\mathcal{I}(s, p) = I(s, p)$;
  - $\mathcal{I}(s, \varphi \circ \psi) = \mathcal{I}(s, \varphi) \,\tilde{\circ}\, \mathcal{I}(s, \psi)$;
  - $\mathcal{I}(s, \forall\alpha) = \mathbf{T}$ iff $\mathcal{I}(\ell, \alpha) = \mathbf{T}$ whenever $\ell(0) = s$;
  - $\mathcal{I}(s, \exists\alpha) = \mathbf{T}$ iff there is some path ℓ starting at s such that $\mathcal{I}(\ell, \alpha) = \mathbf{T}$.
- For path formulas:
  - $\mathcal{I}(\ell, \varphi) = \mathcal{I}\big(\ell(0), \varphi\big)$;
  - $\mathcal{I}(\ell, \alpha \circ \beta) = \mathcal{I}(\ell, \alpha) \,\tilde{\circ}\, \mathcal{I}(\ell, \beta)$;
  - $\mathcal{I}(\ell, \bigcirc\alpha)$ and $\mathcal{I}(\ell, \alpha \mathcal{U}_s \beta)$— as in LTL, evaluated along ℓ as linear timeline.

Examples

- Chess boards.
- Any decision-making process or algorithm.

Examples

- Chess boards.
  - There are reachable positions where White cannot win.
  - White can win.
  - At most one king is in check.
  - If the White king has moved, White won't castle.
- Any decision-making process or algorithm.

Weakening the Logic

- The logic we have introduced is known as CTL^*.
- It is very powerful and highly expressive.

Weakening the Logic

- The logic we have introduced is known as CTL^*.
- It is very powerful and highly expressive.
- Unfortunately, this comes at a price:
- Its decision problem is unusually complex.
- Thus, we would like to limit CTL^* somewhat, while retaining its advantages.

Weakening the Logic

- The logic we have introduced is known as CTL^*.
- It is very powerful and highly expressive.
- Unfortunately, this comes at a price:
- Its decision problem is unusually complex.
- Thus, we would like to limit CTL^* somewhat, while retaining its advantages.
- One popular way is Computation Tree Logic (CTL), which forbids applying temporal operators to anything but state formulas.
  - Applying a temporal operator to a path formula is no longer permitted.

Introducing CTL

The only difference between CTL and CTL^* is in the definition of path formulas.

- In CTL, a path formula is one of $\bigcirc\varphi$, $\varphi\mathcal{U}_{\mathrm{s}}\psi$, where φ and ψ are state formulas.

Introducing CTL

The only difference between CTL and CTL^* is in the definition of path formulas.

- In CTL, a path formula is one of $\bigcirc\varphi$, $\varphi\,\mathcal{U}_{\mathrm{s}}\,\psi$, where φ and ψ are state formulas.
- Therefore, in CTL, temporal operators appear only as part of the compound connectives $\forall\bigcirc$, $\forall\mathcal{U}$, $\forall\Box$, $\forall\Diamond$, $\exists\bigcirc$, $\exists\mathcal{U}$, $\exists\Box$, $\exists\Diamond$.

Introducing CTL

The only difference between CTL and CTL^* is in the definition of path formulas.

- In CTL, a path formula is one of $\bigcirc\varphi$, $\varphi\,\mathcal{U}_{\mathrm{s}}\,\psi$, where φ and ψ are state formulas.
- Therefore, in CTL, temporal operators appear only as part of the compound connectives $\forall\bigcirc$, $\forall\mathcal{U}$, $\forall\Box$, $\forall\Diamond$, $\exists\bigcirc$, $\exists\mathcal{U}$, $\exists\Box$, $\exists\Diamond$.

- For instance, '$\forall\Box\bigcirc\varphi$' is a CTL^* formula, but not a CTL formula.

CTL and CTL*

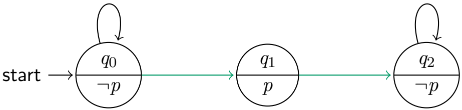
Examples:

- '$\exists\Box\Diamond p$' is not a CTL formula.
- '$\exists\Box\exists\Diamond p$' and '$\exists\Box p$' are CTL formulas..

Examples:

- '$\exists\Box\Diamond p$' is not a CTL formula.
- '$\exists\Box\exists\Diamond p$' and '$\exists\Box p$' are CTL formulas..

- Consider the following interpretation:

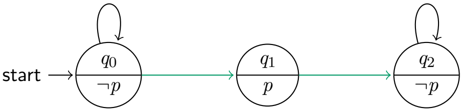


CTL and CTL*

Examples:

- '$\exists\Box\Diamond p$' is not a CTL formula.
- '$\exists\Box\exists\Diamond p$' and '$\exists\Box p$' are CTL formulas..

- Consider the following interpretation:



  - Not a tree, but can be made into one.

Summary

1. Linear-time Temporal Logic
2. Büchi Automata
3. Automata Recognizing Interpretations
4. Extensions of Linear-time Temporal Logic
5. Branching-time Temporal Logic

Time is an illusion. Lunchtime doubly so.
—Douglas Adams

The End.