Expressivness of Temporal Logic

Course in Automatic system Verification: Theory and Applications

Borelli Roberto
borelli.roberto@spes.uniud.it

University of Udine

Abstract

In this presentetion I consider the expressive power of temporal logic

Contents

Background

What is expressive power?

- Let C be a class of structures in a vocabulary L

What is expressive power?

- Let C be a class of structures in a vocabulary L
- We say C is **elementary** (with respect to a logic) if there exists a possibly infinite set of formulae $\tau = \{\tau_1, \tau_2, \tau_3, \dots\}$ such that for each L-structure s it holds $s \in C \iff s \models \tau$

What is expressive power?

- Let C be a class of structures in a vocabulary L
- We say C is **elementary** (with respect to a logic) if there exists a possibly infinite set of formulae $\tau = \{\tau_1, \tau_2, \tau_3, \dots\}$ such that for each L-structure s it holds $s \in C \iff s \models \tau$
- If τ is finite we say C is **base elementary**

Expressive power in classical logic

- As seen in the first part of the course we have several tools to prove that properties are not expressible in first order logic:
  - Ehrenfeucht-Fraïssé games
  - 0/1 laws
  - Locality of first order formulas (Hanf/Gaiffman theorems)
  - Compactness theorem (just for the infinite setting)

[1] FO^k = FO with at most k different variables

Expressive power in classical logic

- As seen in the first part of the course we have several tools to prove that properties are not expressible in first order logic:
  - Ehrenfeucht-Fraïssé games
  - 0/1 laws
  - Locality of first order formulas (Hanf/Gaiffman theorems)
  - Compactness theorem (just for the infinite setting)
- Moving to other logics we have other tools, as a metter of fact we can use Pebble games dealing with FO^k logic[1]

[1] FO^k = FO with at most k different variables

Expressive power in classical logic

- As seen in the first part of the course we have several tools to prove that properties are not expressible in first order logic:
  - Ehrenfeucht-Fraïssé games
  - 0/1 laws
  - Locality of first order formulas (Hanf/Gaiffman theorems)
  - Compactness theorem (just for the infinite setting)
- Moving to other logics we have other tools, as a metter of fact we can use Pebble games dealing with FO^k logic[1]
- Temporal logic ??

[1] FO^k = FO with at most k different variables

Even in FO

- The concept of even is not capturable with FO languages

Even in FO

- The concept of even is not capturable with FO languages
- The language $(aa)^*$ is regular but not star free and hence not first order definable

Even in FO

- The concept of even is not capturable with FO languages
- The language $(aa)^*$ is regular but not star free and hence not first order definable
- We can prove $EVEN_{GRAPHS} = \{G.\ G = (V, E)$ is a graph and $|V|$ is even$\}$ is not FO-expressible with EF games

Even in FO

- The concept of even is not capturable with FO languages
- The language $(aa)^*$ is regular but not star free and hence not first order definable
- We can prove $EVEN_{GRAPHS} = \{G.\ G = (V, E) \text{ is a graph and } |V| \text{ is even}\}$ is not FO-expressible with EF games
  - For each n we consider G_n and H_n
  - G_n is a loop with length 2^n
  - H_n is a loop with length $2^n + 1$

Even in FO

- The concept of even is not capturable with FO languages
- The language $(aa)^*$ is regular but not star free and hence not first order definable
- We can prove $EVEN_{GRAPHS} = \{G.\ G = (V, E)$ is a graph and $|V|$ is even$\}$ is not FO-expressible with EF games
  - For each n we consider G_n and H_n
  - G_n is a loop with length 2^n
  - H_n is a loop with length $2^n + 1$
  - Duplicator always has a winning startegies in $EF_n(G_n, H_n)$ and hence $G_n \equiv_n H_n$

Even in FO

- The concept of even is not capturable with FO languages
- The language $(aa)^*$ is regular but not star free and hence not first order definable
- We can prove $EVEN_{GRAPHS} = \{G.\ G = (V, E) \text{ is a graph and } |V| \text{ is even}\}$ is not FO-expressible with EF games
  - For each n we consider G_n and H_n
  - G_n is a loop with length 2^n
  - H_n is a loop with length $2^n + 1$
  - Duplicator always has a winning startegies in $EF_n(G_n, H_n)$ and hence $G_n \equiv_n H_n$
  - $Gn \in EVEN_{GRAPHS}$ and $H_n \notin EVEN_{GRAPHS}$

$\Longrightarrow$ We do not have a first order formula which defines $EVEN_{GRAPHS}$

Expressive power of LTL

Even in LTL

- $EVEN_{COMPUTATIONS}(p) =$
  $\{\sigma.\ \sigma$ is a computation on which p is true on all even states$\}$

Even in LTL

- $EVEN_{COMPUTATIONS}(p) =$
  $\{\sigma.\ \sigma$ is a computation on which p is true on all even states$\}$
- $EVEN_{COMPUTATIONS}(p)$ is not expressible in LTL

Even in LTL

- $EVEN_{COMPUTATIONS}(p) =$
  $\{\sigma.\ \sigma$ is a computation on which p is true on all even states$\}$
- $EVEN_{COMPUTATIONS}(p)$ is not expressible in LTL
- What about $\psi = p \wedge G(p \rightarrow X\neg p) \wedge G(\neg p \rightarrow Xp)$?

Even in LTL

- $EVEN_{COMPUTATIONS}(p) =$
  $\{\sigma.\ \sigma$ is a computation on which p is true on all even states$\}$
- $EVEN_{COMPUTATIONS}(p)$ is not expressible in LTL
- What about $\psi = p \land G(p \rightarrow X\neg p) \land G(\neg p \rightarrow Xp)$?
- It is also satisfied by the model where p is always true

Even in LTL

- $EVEN_{COMPUTATIONS}(p) =$
  $\{\sigma.\ \sigma$ is a computation on which p is true on all even states$\}$
- $EVEN_{COMPUTATIONS}(p)$ is not expressible in LTL
- What about $\psi = p \land G(p \rightarrow X\neg p) \land G(\neg p \rightarrow Xp)$?
- It is also satisfied by the model where p is always true
- What about $\psi' = \psi \lor Gp$?

Even in LTL

- $EVEN_{COMPUTATIONS}(p) =$
  $\{\sigma.\ \sigma$ is a computation on which p is true on all even states$\}$
- $EVEN_{COMPUTATIONS}(p)$ is not expressible in LTL
- What about $\psi = p \wedge G(p \rightarrow X\neg p) \wedge G(\neg p \rightarrow Xp)$?
- It is also satisfied by the model where p is always true
- What about $\psi' = \psi \vee Gp$?
- Still doesn't cupture the notion of $EVEN_{COMPUTATIONS}(p)$

Even in LTL

- $EVEN_{COMPUTATIONS}(p) =$
  $\{\sigma.\ \sigma$ is a computation on which p is true on all even states$\}$
- $EVEN_{COMPUTATIONS}(p)$ is not expressible in LTL
- What about $\psi = p \wedge G(p \rightarrow X\neg p) \wedge G(\neg p \rightarrow Xp)$?
- It is also satisfied by the model where p is always true
- What about $\psi' = \psi \vee Gp$?
- Still doesn't cupture the notion of $EVEN_{COMPUTATIONS}(p)$
- There are infinite *bad* models of the formula ψ...

Even in LTL

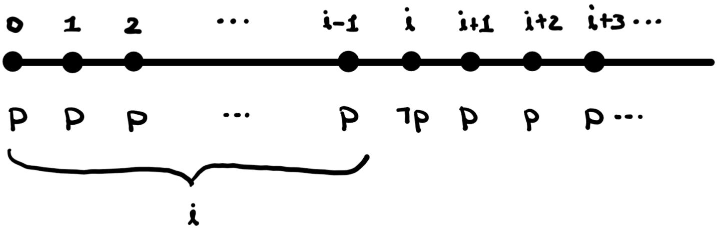
- $EVEN_{COMPUTATIONS}(p) =$
  $\{\sigma.\ \sigma$ is a computation on which p is true on all even states$\}$
- $EVEN_{COMPUTATIONS}(p)$ is not expressible in LTL
- What about $\psi = p \wedge G(p \rightarrow X\neg p) \wedge G(\neg p \rightarrow Xp)$?
- It is also satisfied by the model where p is always true
- What about $\psi' = \psi \vee Gp$?
- Still doesn't cupture the notion of $EVEN_{COMPUTATIONS}(p)$
- There are infinite *bad* models of the formula ψ...
- As a matter of fact we can use infinite formulas of finite length
  $\tau = \{p, XXp, XXXXp, \dots\}$

Even in LTL

To prove the previous result we consider the sequence $p^i(\neg p)p^\omega$

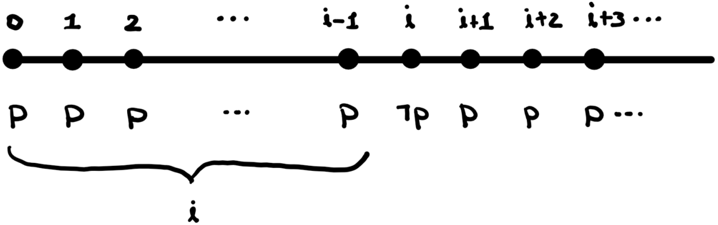
Even in LTL

To prove the previous result we consider the sequence $p^i(\neg p)p^\omega$



Even in LTL

To prove the previous result we consider the sequence $p^i(\neg p)p^\omega$



Theorem

Let $f(p)$ be an $(LTL + P)$ formula,
Let n denote the number of X operators in f
Every sequence $p^i(\neg p)p^\omega$ where $i > n$ has the same truth value on f:
$p^{(n+1)}(\neg p)p^\omega \models f(p) \iff p^{(n+2)}(\neg p)p^\omega \models f(p) \iff \dots$

Even in LTL

Proof.

Denote by $eval_i(f)$ the truth value of f on the sequence $p^i(\neg p)p^\omega$,

Even in LTL

Proof.

Denote by $eval_i(f)$ the truth value of f on the sequence $p^i(\neg p)p^\omega$,we want to prove that $eval_{n+1}(f(p)) = eval_{n+2}(f(p)) = eval_{n+3}(f(p)) = \dots$
In other words we want to prove that $eval_i(f(p))$ is indipendent of $i > n$.

Even in LTL

Proof.

Denote by $eval_i(f)$ the truth value of f on the sequence $p^i(\neg p)p^\omega$,we want to prove that $eval_{n+1}(f(p)) = eval_{n+2}(f(p)) = eval_{n+3}(f(p)) = \dots$
In other words we want to prove that $eval_i(f(p))$ is indipendent of $i > n$.
Consider $j > n$, we proceed by induction on the structure of the formula $f(p)$ (we suppose the theorem holds on all the subformulas):

Even in LTL

Proof.

Denote by $eval_i(f)$ the truth value of f on the sequence $p^i(\neg p)p^\omega$,we want to prove that $eval_{n+1}(f(p)) = eval_{n+2}(f(p)) = eval_{n+3}(f(p)) = \dots$
In other words we want to prove that $eval_i(f(p))$ is indipendent of $i > n$.
Consider $j > n$, we proceed by induction on the structure of the formula $f(p)$ (we suppose the theorem holds on all the subformulas):

- Propositional cases are straightforward.

Even in LTL

Proof.

Denote by $eval_i(f)$ the truth value of f on the sequence $p^i(\neg p)p^\omega$,we want to prove that $eval_{n+1}(f(p)) = eval_{n+2}(f(p)) = eval_{n+3}(f(p)) = \dots$
In other words we want to prove that $eval_i(f(p))$ is indipendent of $i > n$.
Consider $j > n$, we proceed by induction on the structure of the formula $f(p)$ (we suppose the theorem holds on all the subformulas):

- Propositional cases are straightforward.
- Case $f(p) = Xf$. We have $eval_j(Xf) = eval_{j-1}(f)$. f contains $n-1$ X opertors so we have $j-1 > n-1$ and hence by the inductive hypothesis the value of $eval_{j-1}(f)$ is indipendent of j.

Even in LTL

Proof.

Denote by $eval_i(f)$ the truth value of f on the sequence $p^i(\neg p)p^\omega$,we want to prove that $eval_{n+1}(f(p)) = eval_{n+2}(f(p)) = eval_{n+3}(f(p)) = \dots$
In other words we want to prove that $eval_i(f(p))$ is indipendent of $i > n$.
Consider $j > n$, we proceed by induction on the structure of the formula $f(p)$ (we suppose the theorem holds on all the subformulas):

- Propositional cases are straightforward.
- Case $f(p) = Xf$. We have $eval_j(Xf) = eval_{j-1}(f)$. f contains $n-1$ X opertors so we have $j-1 > n-1$ and hence by the inductive hypothesis the value of $eval_{j-1}(f)$ is indipendent of j.

Even in LTL

Proof.

- Case $f(p) = f_1 \ U \ f_2$.

Even in LTL

Proof.

- Case $f(p) = f_1 \; U \; f_2$. From
  - $(f_1 \; U \; f_2) = f_2 \vee (f_1 \wedge X(f_1 \; U \; f_2))$
  - $eval_j(Xf) = eval_{j-1}(f)$

Even in LTL

Proof.

- Case $f(p) = f_1 \ U \ f_2$. From
  - $(f_1 \ U \ f_2) = f_2 \vee (f_1 \wedge X(f_1 \ U \ f_2))$
  - $eval_j(Xf) = eval_{j-1}(f)$

  we have $eval_j(f_1 \ U \ f_2) = eval_j(f_2) \vee (eval_j(f_1) \wedge eval_{j-1}(f_1 \ U \ f_2))$

Even in LTL

Proof.

- Case $f(p) = f_1 \; U \; f_2$. From
  - $(f_1 \; U \; f_2) = f_2 \vee (f_1 \wedge X(f_1 \; U \; f_2))$
  - $eval_j(Xf) = eval_{j-1}(f)$

  we have $eval_j(f_1 \; U \; f_2) = eval_j(f_2) \vee (eval_j(f_1) \wedge eval_{j-1}(f_1 \; U \; f_2))$ and unfolding n times:
  $eval_j(f_1 \; U \; f_2) = eval_j(f_2) \vee (eval_j(f_1) \wedge eval_{j-1}(f_2) \vee (eval_{j-1}(f_1) \wedge \cdots \wedge (eval_{n+1}(f_2) \vee (eval_{n+1}(f_1) \wedge eval_n(f_1 \; U \; f_2))) \ldots))$

Even in LTL

Proof.

- Case $f(p) = f_1 \ U \ f_2$. From
  - $(f_1 \ U \ f_2) = f_2 \vee (f_1 \wedge X(f_1 \ U \ f_2))$
  - $eval_j(Xf) = eval_{j-1}(f)$

  we have $eval_j(f_1 \ U \ f_2) = eval_j(f_2) \vee (eval_j(f_1) \wedge eval_{j-1}(f_1 \ U \ f_2))$ and unfolding n times:
  $eval_j(f_1 \ U \ f_2) = eval_j(f_2) \vee (eval_j(f_1) \wedge eval_{j-1}(f_2) \vee (eval_{j-1}(f_1) \wedge \cdots \wedge (eval_{n+1}(f_2) \vee (eval_{n+1}(f_1) \wedge eval_n(f_1 \ U \ f_2))) \ldots))$
  By the inductive hypothesis
  $eval_j(f_k) = eval_{j-1}(f_k) = \cdots = eval_{n+1}(f_k)$ for $k = 1, 2$

Even in LTL

Proof.

- Case $f(p) = f_1 \ U \ f_2$. From
  - $(f_1 \ U \ f_2) = f_2 \lor (f_1 \land X(f_1 \ U \ f_2))$
  - $eval_j(Xf) = eval_{j-1}(f)$

  we have $eval_j(f_1 \ U \ f_2) = eval_j(f_2) \lor (eval_j(f_1) \land eval_{j-1}(f_1 \ U \ f_2))$
  and unfolding n times:
  $eval_j(f_1 \ U \ f_2) = eval_j(f_2) \lor (eval_j(f_1) \land eval_{j-1}(f_2) \lor (eval_{j-1}(f_1) \land \cdots \land (eval_{n+1}(f_2) \lor (eval_{n+1}(f_1) \land eval_n(f_1 \ U \ f_2))) \ldots))$
  By the inductive hypothesis
  $eval_j(f_k) = eval_{j-1}(f_k) = \cdots = eval_{n+1}(f_k)$ for $k = 1, 2$
  so we have:
  $eval_j(f_1 \ U \ f_2) = eval_{n+1}(f_2) \lor (eval_{n+1}(f_1) \land eval_n(f_1 \ U \ f_2))$

Even in LTL

Proof.

- Case $f(p) = f_1 \; U \; f_2$. From
  - $(f_1 \; U \; f_2) = f_2 \vee (f_1 \wedge X(f_1 \; U \; f_2))$
  - $eval_j(Xf) = eval_{j-1}(f)$

we have $eval_j(f_1 \; U \; f_2) = eval_j(f_2) \vee (eval_j(f_1) \wedge eval_{j-1}(f_1 \; U \; f_2))$
and unfolding n times:
$eval_j(f_1 \; U \; f_2) = eval_j(f_2) \vee (eval_j(f_1) \wedge eval_{j-1}(f_2) \vee (eval_{j-1}(f_1) \wedge \cdots \wedge (eval_{n+1}(f_2) \vee (eval_{n+1}(f_1) \wedge eval_n(f_1 \; U \; f_2))) \ldots))$
By the inductive hypothesis
$eval_j(f_k) = eval_{j-1}(f_k) = \cdots = eval_{n+1}(f_k)$ for $k = 1, 2$
so we have:
$eval_j(f_1 \; U \; f_2) = eval_{n+1}(f_2) \vee (eval_{n+1}(f_1) \wedge eval_n(f_1 \; U \; f_2))$ and hence
$eval_{n+1}(f_1 \; U \; f_2) = eval_{n+2}(f_1 \; U \; f_2) = eval_{n+3}(f_1 \; U \; f_2) = \ldots$

$\square$

Even in LTL

Corollary

For each $m \geq 2$ the set $m_{COMPUTATIONS}(p) =$ $\{\sigma.\ \sigma$ is a computation on which p is true at each state multiple of $m\}$ is not expressible in LTL

Even in LTL

Corollary

For each $m \geq 2$ *the set* $m_{COMPUTATIONS}(p) =$
$\{\sigma.\ \sigma$ *is a computation on which p is true at each state multiple of m*$\}$ *is not expressible in LTL*

Proof.

Suppose that the formula $f(p)$ with n X operators captures $m_{COMPUTATIONS}(p)$ for a given m.

Even in LTL

Corollary

For each $m \geq 2$ *the set* $m_{COMPUTATIONS}(p) =$
$\{\sigma.\ \sigma$ *is a computation on which* p *is true at each state multiple of* $m\}$ *is not expressible in LTL*

Proof.

Suppose that the formula $f(p)$ with n X operators captures $m_{COMPUTATIONS}(p)$ for a given m. Choose k such that $km > n$. Then we have

Even in LTL

Corollary

For each $m \geq 2$ the set $m_{COMPUTATIONS}(p) =$ $\{\sigma.\ \sigma$ is a computation on which p is true at each state multiple of $m\}$ is not expressible in LTL

Proof.

Suppose that the formula $f(p)$ with n X operators captures $m_{COMPUTATIONS}(p)$ for a given m. Choose k such that $km > n$. Then we have

- $p^{km+1}(\neg p)p^{\omega} \in m_{COMPUTATIONS}(p)$
- $p^{km}(\neg p)p^{\omega} \notin m_{COMPUTATIONS}(p)$

Even in LTL

Corollary

*For each $m \geq 2$ the set $m_{COMPUTATIONS}(p) =$
$\{\sigma.\ \sigma$ is a computation on which p is true at each state multiple of $m\}$ is not expressible in LTL*

Proof.

Suppose that the formula $f(p)$ with n X operators captures $m_{COMPUTATIONS}(p)$ for a given m. Choose k such that $km > n$. Then we have

- $p^{km+1}(\neg p)p^{\omega} \in m_{COMPUTATIONS}(p)$
- $p^{km}(\neg p)p^{\omega} \notin m_{COMPUTATIONS}(p)$

but $eval_{km+1}(f(p)) = eval_{km}(f(p))$ □

LTL and FO

The previuous result was not a case, infact we have an equivalence between LTL and FO theory of discrete linear orders (Kamp Theorem)

LTL and FO

The previuous result was not a case, infact we have an equivalence between LTL and FO theory of discrete linear orders (Kamp Theorem)

Theorem

$(LTL + P)[XU, XS] \subseteq FO^3[<]$

XU = Strong Strict Until, XS = Strong Strict Since

LTL and FO

The previuous result was not a case, infact we have an equivalence between LTL and FO theory of discrete linear orders (Kamp Theorem)

Theorem

$(LTL + P)[XU, XS] \subseteq FO^3[<]$

XU = Strong Strict Until, XS = Strong Strict Since

Theorem

$FO[<] \subseteq (LTL + P)[XU, XS]$

LTL and FO

The previuous result was not a case, infact we have an equivalence between LTL and FO theory of discrete linear orders (Kamp Theorem)

Theorem

$(LTL + P)[XU, XS] \subseteq FO^3[<]$

XU = Strong Strict Until, XS = Strong Strict Since

Theorem

$FO[<] \subseteq (LTL + P)[XU, XS]$

Corollary

$FO[<] = FO^3[<]$

Express EVEN in LTL

- To express EVEN in LTL the author proposed 2 solutions

Express EVEN in LTL

- To express EVEN in LTL the author proposed 2 solutions
- First, if we add quantifiers to LTL we could use the formula $\psi(p) = \exists q(q \land G(q \rightarrow X\neg q) \land G(\neg q \rightarrow Xq) \land G(q \rightarrow p))$

Express EVEN in LTL

- To express EVEN in LTL the author proposed 2 solutions
- First, if we add quantifiers to LTL we could use the formula $\psi(p) = \exists q(q \land G(q \rightarrow X\neg q) \land G(\neg q \rightarrow Xq) \land G(q \rightarrow p))$
- Another way is to add the operator $even(p)$ in the syntax of LTL

Express EVEN in LTL

- To express EVEN in LTL the author proposed 2 solutions
- First, if we add quantifiers to LTL we could use the formula $\psi(p) = \exists q(q \wedge G(q \rightarrow X\neg q) \wedge G(\neg q \rightarrow Xq) \wedge G(q \rightarrow p))$
- Another way is to add the operator $even(p)$ in the syntax of LTL
- but then the question is... which operators can we add to LTL to increase its expressive power while keeping its decision problem PSPACE-complete?

Express EVEN in LTL

- To express EVEN in LTL the author proposed 2 solutions
- First, if we add quantifiers to LTL we could use the formula $\psi(p) = \exists q(q \land G(q \to X\neg q) \land G(\neg q \to Xq) \land G(q \to p))$
- Another way is to add the operator $even(p)$ in the syntax of LTL
- but then the question is... which operators can we add to LTL to increase its expressive power while keeping its decision problem PSPACE-complete?
- It turns out that operators corresponding to any property definable by a right-linear grammar (or in other words regular properties) can be added to LTL while not increasing the complexity of its decision procedure

Express EVEN in LTL

- To express EVEN in LTL the author proposed 2 solutions
- First, if we add quantifiers to LTL we could use the formula $\psi(p) = \exists q(q \wedge G(q \rightarrow X\neg q) \wedge G(\neg q \rightarrow Xq) \wedge G(q \rightarrow p))$
- Another way is to add the operator $even(p)$ in the syntax of LTL
- but then the question is... which operators can we add to LTL to increase its expressive power while keeping its decision problem PSPACE-complete?
- It turns out that operators corresponding to any property definable by a right-linear grammar (or in other words regular properties) can be added to LTL while not increasing the complexity of its decision procedure
- The logic obtained in this way is called ETL (Extended Temporal Logic)

Conclusion

Concluding remarks

- Often we observe that more expressive power comes at a cost

Bibliography I

Diekert, V. and P. Gastin (Jan. 2008). "First-order definable languages". In: pp. 261–306.

Kamp, H. (1968). "Tense Logic and the Theory of Linear Order". PhD thesis. Ucla.

Wolper, P. (1983). "Temporal logic can be more expressive". In: *Information and Control* 56.1, pp. 72–99. ISSN: 0019-9958. DOI: https://doi.org/10.1016/S0019-9958(83)80051-5. URL: https://www.sciencedirect.com/science/article/pii/S0019995883800515.