Lógica para Computación

Marcelo F. Frias
ITBA y CONICET

Contenidos del Curso

- Introducción a la lógica proposicional y a la lógica de primer orden.
- Lógica relacional.
- Especificación de programas: precondición más débil, corrección parcial y corrección total.
- Java Modeling Language (JML): un lenguaje para la especificación de programas.
- Detección automática de fallas.
- Reparación automática de fallas.

Para qué?

- Para qué nos interesa la lógica a los computólgos?
- Nos interesa?
- Es sólo un ejercicio intelectual para iluminados?
- Tiene alguna utilidad práctica?

Lógica Proposicional

- Las variables p1, p2,…,pn,… representan proposiciones, i.e., propiedades sobre las cuales se puede realizar un juicio sobre su veracidad.
    - Ejemplo:
        - Los pingüinos usan frac
        - Los gatos tienen ojos verdes
        - Los perros son animales

Lógica Proposicional

- A partir de las variables proposicionales (fórmulas atómicas) se pueden construir fórmulas complejas de la siguiente forma:

- si A y B son fórmulas proposicionales, **!A** (se lee no A), **(A && B)** (se lee A y B) y **(A || B)** (se lee A o B) son fórmulas proposicionales.

Lógica Proposicional

- Dada una fórmula proposicional podemos preguntarnos su valor de verdad.

- El siguiente algoritmo

  T : Val x Fórmulas Proposicionales —> {t, f},

  conocido como “Tablas de Verdad” nos permite analizar el valor de verdad de una fórmula proposicional en todas las valuaciones de las proposiciones.

Lógica Proposicional

- Val : {p1,…,pn,…} —> {t,f} es una valuación.
- T_Val(pi) = Val(pi)
- T_Val(!A) = t ssi T_Val(A) = f
- T_Val(A && B) = T_Val(A) ^ T_Val(B)
- T_Val(A || B) = T_Val(A) v T_Val(B)

Lógica Proposicional

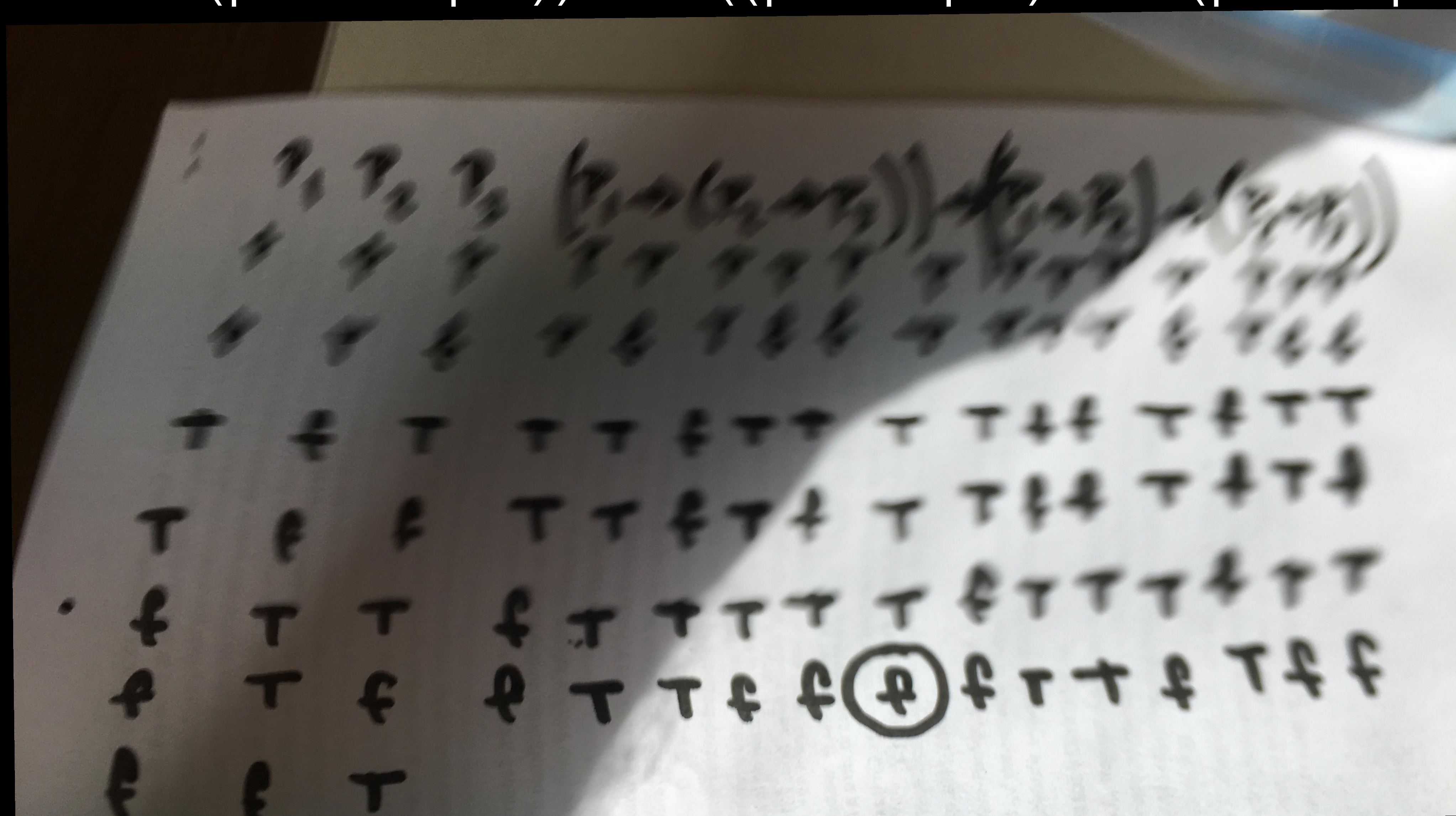
- Definición: Una fórmula F es una tautología, si para toda valuación Val, T_Val(F) = t.

- Teorema (Decidibilidad de la Lógica Proposicional): Hay un algoritmo (conocido como Tablas de Verdad), que permite para cada fórmula determinar si la misma es una tautología.

Lógica Proposicional

- Ejemplo:

(p1 => (p2 => p3)) => ((p1=>p2) => (p2=>p3))



Lógica Proposicional

- Ejercicio:

(p1 => (p2 => p3)) => ((p1=>p2) => (p1=>p3))

Pregunta: Por qué es correcto restringirse sólo a las variables proposicionales que aparecen en la fórmula?

Lógica Clásica de Primer Orden

- Permite expresar propiedades sobre objetos. Las evaluación del valor de verdad depende de un dominio semántico mucho más rico que las valuaciones de la lógica proposicional.

Ejemplo 1: Todos los peces tienen agallas.

Ejemplo 2: Nemo tiene una aleta más pequeña

Lógica Clásica de Primer Orden: Sintaxis

- Vamos a tener términos que denotan objetos (es decir, un término será una expresión sintáctica que al interpretarla semánticamente referirá a un objeto como Nemo, Dory, etc…)
- Vamos a tener fórmulas que nos permitan escribir propiedades sobre los objetos.
    - es_amigo_de(Nemo, Dory)
    - es_hijo_de(Nemo, Merlín)
    - es_un_pez_payaso(Nemo)
    - ParaTodo x ((es_un_pez_payaso(x) && es_hijo_de(x, Merlín)) =>

      es_amigo_de(x, Dory))

Lógica Clásica de Primer Orden: Lenguaje de Primer Orden

- Un lenguaje de primer orden es una estructura <C, F, P> tal que

  - C es un conjunto de *símbolos de constante,* C = {c1, c2,…, cn}

  - F es un conjunto de *símbolos de función*. Cada símbolo tiene una aridad (tipado) determinada. F = {f1_{t_1}, f2_{t_2},…,fk_{t_k}}.

  - P es un conjunto de *símbolos de predicado*. Cada símbolo tiene una aridad determinada. P = {p1_{r_1}, p2_{r_2},…,pm_{r_m}}.

Lógica Clásica de Primer Orden: Lenguaje de Primer Orden

IMPORTANTE: Los nombres de los símbolos de constantes, de los símbolos de función y de los símbolos de predicados no significan nada. Será cuando agreguemos axiomas que tomarán un significado más preciso. O cuando los mapeemos en una estructura de primer orden.

- Un le… na idea de sobre… oder habla…
  - L_N… {0}, {suc_1}, {<=_2, >_2, >=0_1}>.
  - L_N- = <{0}, {pred_1}, {<=, >, >=0}>.

Lógica Clásica de Primer Orden: Términos

- Como dijimos antes, los términos denotan objetos. Vamos a ver las reglas que nos permiten determinar si una expresión es un término bien formado (correcto).

- Sea L = <C, F, P> un lenguaje de primer orden. Sea V = {v1,…,vk,…} un conjunto infinito (numerable) de variables. Intuitivamente, las variables son términos que denotan objetos arbitrarios.

    - Si t in V, entonces t es un término.

    - Si t in C, entonces t es un término.

    - Si f in F tiene aridad n y t1,…,tn son términos, f(t1,…,tn) es un término.

    - Nada más es un término.

Lógica Clásica de Primer Orden: Términos

- Para el lenguaje de primer orden <{0}, {suc}, {<=, >, >=0}>, por ejemplo tenemos v3, 0, suc(0), suc(v5), suc(suc(suc(0)))…

- Para el lenguaje de primer orden <{0}, {pred}, {<=, >, >=0}>, por ejemplo tenemos v3, 0, pred(0), pred(v5), pred(pred(pred(0)))…

Lógica Clásica de Primer Orden: Fórmulas

- Mientras que los términos denotan objetos, las fórmulas tienen un valor de verdad.
- Dado un lenguaje de primer orden L = <C, F, P>, si p in P de aridad n y t1, …,tn son términos, entonces P(t1,…,tn) es una fórmula atómica (porque es el tipo más pequeño de fórmula).
- Si f es una fórmula atómica, entonces f es una fórmula.
- si f es una fórmula, !f es una fórmula
- si f1 y f2 son fórmulas, f1 && f2, f1 || f2 y f1=>f2 son fórmulas.
- si f es una fórmula, y v es una variable, (forall v f) y (some v f) son fórmulas (cuantificación universal y existencial).
- Nada más es fórmula.

Lógica Clásica de Primer Orden: Fórmulas

- forall v1 (forall v2 (>(suc(v1), suc(v2)) => >(v1, v2)))
- Equivalentemente podemos usar (cuando es apropiado) los predicados infijos: forall v1 (forall v2 (suc(v1) > suc(v2) => v1 > v2))
- forall v1 (forall v2 (suc(v1) > suc(v2) => v1 <= v2))

Lógica Clásica de Primer Orden: Estructuras Adecuadas

- Dado un lenguaje de primer orden $L = \langle C, F, P \rangle$, una estructura adecuada para L es una estructura $A\_L = \langle \mathbf{A}, \mathbf{C}, \mathbf{F}, \mathbf{P} \rangle$ tal que
    - **A** es un conjunto no vacío de objetos.
    - Si $C = \{c1,\ldots,ck\}$, $\mathbf{C} = \{\mathbf{c}1,\ldots,\mathbf{c}k\} \subseteq \mathbf{A}$ es un conjunto de *constantes*.
    - Si $F = \{f1,\ldots,fn\}$, $\mathbf{F} = \{\mathbf{f}1,\ldots,\mathbf{f}n\}$ es una familia de funciones sobre **A** de la aridad correspondiente. I.e., si fj es r-aria, $\mathbf{f}j : \mathbf{A}^r \rightarrow \mathbf{A}$.
    - Si $P = \{p1,\ldots,pl\}$, $\mathbf{P} = \{\mathbf{p}1,\ldots,\mathbf{p}l\}$ es una familia de relaciones sobre **A** de la aridad correspondiente. I.e., si pj es s-aria, $\mathbf{p}j \subseteq \mathbf{A}^s$.

Lógica Clásica de Primer Orden: Estructuras Adecuadas

- Ejemplo: Para el lenguaje de primer orden L_N+ = <{0}, {suc}, {<=, >, >=0}>, podemos tener <N, {cero}, {+1}, {menor_o_igual, mayor, mayor_o_igual que cero}>.

Lógica Clásica de Primer Orden: Presentación de Teoría

- Dado un lenguaje de primer-orden L y un conjunto de fórmulas de primer order sobre L \Gamma, el par <L, \Gamma> es una presentación de teoría.

- La teoría es el conjunto de fórmulas sobre el lenguaje L que son consecuencia de las fórmulas en \Gamma (a definir a continuación).

Lógica Clásica de Primer Orden: Consecuencia

- Sea L = <C,F,P> un lenguaje de primer orden, sea \mathcal{A} = <A,**C**,**F**,**P**> una estructura adecuada para L. Una valuación de las variables es una función \mu : Var \to A.

- A partir de la noción de valuación podemos ver el valor de un término dentro de una estructura. Definimos:

  - Sea v in Var. [[v]]_\mu = \mu(v).

  - Sea c in C, [[c]]_\mu = **c**.

  - Sea f in F con aridad k, y t1,…,tk términos. [[f(t1,…,tk)]]_\mu = **f**([[t1]]_\mu,…,[[tk]]_\mu).

Lógica Clásica de Primer Orden: Consecuencia

- Una vez definido el valor de un término dados una estructura adecuada y una valuación, podemos evaluar el valor de verdad de una fórmula.
- Sea $\mathcal{A} = <A, **C**, **F**, **P**>$ una estructura adecuada para un lenguaje de primer orden $L = <C, F, P>$. Sea $\mu : \mathit{Var} \to A$ una valuación. Definimos:
    - $\mathcal{A} |= p(t1,\ldots,tn) [\mu]$ si $<[[t1]]_\mu,\ldots,[[tn]]_\mu> \in **p**$.
    - $\mathcal{A} |= !\alpha[\mu]$ iff not $\mathcal{A} |= \alpha[\mu]$.
    - $\mathcal{A} |= (\alpha || \beta) [\mu]$ iff $\mathcal{A} |= \alpha [\mu]$ or $\mathcal{A} |= \beta [\mu]$.
    - $\mathcal{A} |= (\alpha && \beta) [\mu]$ iff $\mathcal{A} |= \alpha [\mu]$ and $\mathcal{A} |= \beta [\mu]$.
    - $\mathcal{A} |= \exists x : \alpha [\mu]$ ssi there is $a\inA$ tal que $\mathcal{A} |= \alpha [\mu x/a]$
    - $\mathcal{A} |= \forall x : \alpha [\mu]$ ssi para todo $a\inA$, $\mathcal{A} |= \alpha [\mu x/a] $

Lógica Clásica de Primer Orden: Consecuencia

- Dado un conjunto de fórmulas $\Gamma = {\gamma_1,\ldots,\gamma_k}$, definimos $ \mathcal{A} \models \Gamma[\mu]$ ssi $ \mathcal{A} \models \gamma_i[\mu]$ para todo $1\leq i \leq k$.

- α es consecuencia de Γ ($ \Gamma |= \alpha$) ssi para toda estructura $ \mathcal{A}$ y toda valuación μ, vale $ \mathcal{A} |= \Gamma[\mu]$ implies $ \mathcal{A} |= \alpha[\mu]$.

Lógica Clásica de Primer Orden: Modelo

- Dado un conjunto de fórmulas Γ una estructura adecuada $\mathcal{A}$ se dice \emph{modelo} si para toda valuación μ $ \mathcal{A} |= \Gamma[\mu]$.

Metateoremas Importantes de la Lógica de Primer Orden

- Upward Löwenheim-Skolem: Si Γ tiene modelos finitos arbitrariamente grandes o al menos un modelo infinito, entonces tiene modelos de todos los cardinales infinitos.
  - Es posible escribir un conjunto de axiomas que caracterice a los números naturales unívocamente?

Metateoremas Importantes de la Lógica de Primer Orden

- Downward Löwenheim-Skolem: Si Γ tiene un modelo infinito, entonces tiene un modelo numerable.
  - Es posible caracterizar de forma unívoca a los números reales en primer orden?

Metateoremas Importantes de la Lógica de Primer Orden

- Compacidad: Γ es satisfacible ssi es finitamente satisfacible (Γ puede ser infinito).
- Indecidibilidad: No existe un algoritmo que permita determinar si una fórmula de primer orden es universalmente válida (A. Church)

Clausura (reflexo-)transitiva

- Cómo hacemos para especificar que una lista simplemente enlazada no tiene un ciclo?
    - $\forall n (*next(head,n) => !*next(next(n),n))$
- Sin utilizar clausura no es posible hacerlo de forma declarativa. Ahora… es la clausura expresable en primer orden?
- Si lo fuera, podemos escribir:
    - $\forall n (*suc(0,n))$.
    - Qué dice la fórmula de arriba?