# Ingeniería del Software II
# 7 - Especificación de Sistemas

# Especificaciones de Sistemas

Varios enfoques tradicionales para la construcción formal y la verificación de sistemas se basan en el uso de especificaciones.

Una especificación puede usarse para varias tareas, tales como:

- descubrir inconsistencias (por ejemplo, requerimientos contradictorios) o falta de detalles en las funcionalidades requeridas del sistema a construir;
- análisis de propiedades del sistema a construir antes de comenzar con su construcción;
- verificación del sistema construido con respecto a su especificación;
- construcción rigurosa de implementaciones a partir de especificaciones.

# Características de los Lenguajes de Especificación

Las especificaciones son descripciones abstractas de sistemas.

Se escriben en algún lenguaje de especificación.

Existe una gran variedad de lenguajes de especificación, clasificables de acuerdo a varias características.

La elección del lenguaje de especificación depende de tales características

# Características de los Lenguajes de Especificación

Propiedades deseables de los lenguajes de especificación:

- **formal**: no da lugar a ambigüedades => no requiere esfuerzos adicionales para evitar confusiones e interpretaciones erróneas de la especificación;
- **declarativo**: facilita la caracterización del sistema a construir y de las propiedades a analizar (se enfoca en el "qué");
- **expresivo**: permite describir una amplia gama de características de los sistemas a construir y de las propiedades a analizar;
- que posea **mecanismos efectivos para el análisis** de propiedades: demanda menos esfuerzos para el estudio de las consecuencias de las funcionalidades requeridas del sistema a construir.

# Las Lógicas como Lenguajes de Especificación

Cuando la intención principal es analizar propiedades de los sistemas a construir, es crucial contar con mecanismos efectivos para el análisis.

Un enfoque usado con frecuencia es el de utilizar lógicas que se ajusten a la descripción de las propiedades de interés como lenguajes de especificación.

Las lógicas como lenguajes de especificación tienen la particularidad de ser formales, declarativas, y además cuentan con un mecanismo de análisis de propiedades: la deducción.

Con frecuencia, es necesario buscar un balance entre poder expresivo y efectividad para el análisis.

# Lógica Proposicional

Es una de las lógicas más simples.

Como lenguaje de especificación, posee las siguientes características:

- poder expresivo limitado.
- importantes propiedades metalógicas (consistencia, completitud, decidibilidad, etc.) y variedad de cálculos de prueba (cálculo de secuentes, deducción natural, deducción "a la Hilbert", tableaux, etc.) que pueden usarse para el análisis (demostración de propiedades).
- determinar satisfactibilidad y validez es decidible, i.e., pueden verificarse automáticamente.

# Lógica Proposicional

## Sintaxis de la Lógica Proposicional

- variables proposiciones: $p$, $q$, $r$, ...
- conectivos lógicos: $\wedge$, $\vee$, $\neg$, $\Rightarrow$, ...
- fórmulas: $p \wedge q$, $\neg r$, ...

Dado que los conectivos son fijos, el conjunto de fórmulas está determinado por las proposiciones atómicas.

# Lógica Proposicional

## Especificaciones en Lógica Proposicional

- Las proposiciones son el elemento básico de especificación.
- Las proposiciones representan sentencias atómicas, que pueden ser verdaderas o falsas.
- Los conectivos lógicos permiten describir sentencias compuestas a partir de sentencias básicas.

# Lógica Proposicional

## Semántica

Dado una teoría proposicional (i.e., un conjunto de fórmulas proposicionales incluyendo los axiomas de la lógica proposicional), todo modelo de ésta puede verse como una función

$$I : \mathcal{PA} \rightarrow \{true, false\}$$

En particular, diremos que

una fórmula proposicional $\phi$ se satisface en $I$ si $I \models \phi$.

Esta última noción se define inductivamente como sigue:

$$\begin{array}{lll} I \models p & \text{sii} & I(p) = true \\ I \models \neg\phi & \text{sii} & I \not\models \phi \\ I \models \phi \wedge \psi & \text{sii} & I \models \phi \text{ y } I \models \psi \end{array}$$

# Lógica Proposicional

**Semántica (cont.)**

El símbolo $\models$ con frecuencia se utiliza para representar **deducción**.

Esto es, $\models$ denota una relación entre un conjunto de fórmulas y una fórmula.

Si $\Gamma$ es un conjunto de fórmulas proposicionales,

$\Gamma \models \phi$ sii para toda interpretación $I$, $I \models \Gamma$ implica $I \models \phi$

donde $I \models \Gamma$ sii $\forall \psi \in \Gamma : I \models \psi$

# Lógica Proposicional

## Algunas propiedades metalógicas

**Lema de la Deducción** (a nivel semántico):

$$\Gamma, \phi \models \psi \quad \Leftrightarrow \quad \Gamma \models \phi \Rightarrow \psi$$

**Decidibilidad de satisfacción:**

dada una fórmula $\phi$, existe un algoritmo que permite decidir si existe algún modelo $I$ tal que

$$I \models \phi$$

o no.

# SAT Solving en Lógica Proposicional

- Usando las propiedades metalógicas anteriores, se puede usar la técnica denominada "**SAT Solving**" para realizar deducciones.
- Supongamos que queremos saber si una fórmula $\phi$ es consecuencia de ciertas premisas $\Gamma$. Es decir, queremos saber si es cierto que:

$$\Gamma \models \phi$$

- Gracias al Lema de la Deducción, podemos decir que

$$\Gamma \models \phi \quad \text{sii} \quad \text{no existe } I \text{ tal que } I \models \neg \left( \left( \bigwedge_{\psi \in \Gamma} \psi \right) \Rightarrow \phi \right)$$

- Para decidir esto último podemos usar el algoritmo de decidibilidad de la satisfacción.

# SAT Solving en Lógica Proposicional

**Ventajas y desventajas de reducir deducción a un problema de satisfacción:**

- **Ventaja fundamental**: se obtiene una técnica absolutamente automática para decidir si un enunciado es consecuencia de un conjunto de premisas.
- **Desventaja**: complejidad temporal de este enfoque.
  - el algoritmo de decisión de satisfacción en lógica proposicional es $O(n \times 2^n)$, donde $n$ representa el número de proposiciones en la fórmula.
  - se sabe que el problema en sí mismo es NP-completo.

# Lógicas de Primer Orden

- La lógica proposicional tiene un poder expresivo limitado.

  => poca utilidad como lenguaje de especificación

- Pueden considerarse como alternativas más expresivas algunas lógicas de primer orden.
- Problema: el problema de satisfacción de fórmulas no es decidible en lógica de primer orden.

  => no se puede usar como algoritmo de demostración la reducción de deducción a un problema de satisfacción.

- Existen, sin embargo, cálculos de prueba completos para lógica de primer orden.

  => Asistencia con demostradores de teoremas y verificadores de demostración.

# Lógicas de Primer Orden

**Sintaxis:**

variables: $x$, $y$, $z$, ...

símbolos de función: $f(x, y)$, $g(x, y, z)$, $c$, ...

predicados y relaciones: $p$, $q(x)$, $r(s, t)$, ...

conectivos y cuantificadores: $\forall$, $\exists$, $\wedge$, $\vee$, $\neg$, $\Rightarrow$ ...

fórmulas: $p \wedge r(c, h(c))$, $\forall x : \exists y : q(x) \Rightarrow r(c, y)$, ...

# Lógicas de Primer Orden

**Semántica**:

- las interpretaciones requieren un dominio
- los símbolos de función se interpretan como funciones y los de predicados como relaciones sobre el dominio de la interpretación
- las variables se interpretan mediante asignaciones.
- los conectivos lógicos tienen una interpretación fija:
  - $\phi \wedge \psi$ se hace verdadera si ambas $\phi$ y $\psi$ se hacen verdadera,
  - $\neg\phi$ se hace verdadera cada vez que $\phi$ no lo es,
  - etc.
- $\forall$ tiene una interpretación fija: requiere que la fórmula en cuestión sea verdadera para todas las posibles asignaciones a la variable ligada al cuantificador.

# Lógicas de Primer O

La semántica formal la vieron en Lógica el cuatrimestre pasado

**Semántica**:

- las interpretaciones requieren un dominio
- los símbolos de función se interpretan como funciones y los de predicados como relaciones sobre el dominio de la interpretación
- las variables se interpretan mediante asignaciones.
- los conectivos lógicos tienen una interpretación fija:
  - $\phi \wedge \psi$ se hace verdadera si ambas $\phi$ y $\psi$ se hacen verdadera,
  - $\neg\phi$ se hace verdadera cada vez que $\phi$ no lo es,
  - etc.
- $\forall$ tiene una interpretación fija: requiere que la fórmula en cuestión sea verdadera para todas las posibles asignaciones a la variable ligada al cuantificador.

# SAT Solving en Lógica de Primer Orden

- El problema de satisfacción es indecidible en LPO
- De todas maneras es útil aplicar una técnica similar a la aplicada para lógica proposicional.
- La cantidad de interpretaciones posibles para fórmulas en lógica de primer orden puede ser infinita.
  - las variables pueden tomar valores arbitrarios de sus dominios, y los dominios pueden ser infinitos.

# SAT Solving en Lógica de Primer Orden

**Sin embargo**:

- si ponemos una cota k en el número máximo de elementos en el dominio de una interpretación, podemos examinar exhaustivamente todas las interpretaciones posibles de "tamaño" a lo sumo k.
- si encontramos un modelo de la fórmula en cuestión, entonces dicha fórmula es satisfactible.
- si no encontramos modelo, sólo podemos garantizar que no existen modelos de la fórmula de "tamaño" menor o igual que k.

# SAT Solving en Lógica de Primer Orden

**Un ejemplo:**

Consideremos la formula

$$\forall x : p(x, c) \wedge q(f(x, x))$$

$5 \times 2^5 \times 2^{(5\times5)} \times 5^{(5\times5)}$

Esta fórmula tiene:

- una función binaria ( $f$ )
- una constante ( $c$ )
- un predicado de aridad uno y otro de dos ( $q$ y $p$ respectivamente)

¿Cuántas interpretaciones posibles de "tamaño" 5 existen?

# Lógicas de Primer Orden

**Poder expresivo**:

- La lógica de primer orden es un formalismo muy expresivo
- Permite especificar gran cantidad de propiedades (en el contexto de especificaciones de sistemas de software).
- Sin embargo, aún hay muchas cosas que no puede expresar.
- Por ejemplo:
  - clausura transitiva de una relación.
  - en particular no puede expresar alcanzabilidad en un grafo.

# Álgebra de relaciones

El álgebra de relaciones es una extensión de un algebra de Boole con operaciones que permiten manipular relaciones binarias.

**Constantes**:

- **none** es la *relación vacía*
- **iden** es la *relación de identidad*
- **univ** es la *relación universal*

**Operadores Unarios**:

- $\overline{R}$ es el *complemento* de $R$
- $\tilde{R}$ es la *conversa* de $R$
- $R^+$ es la *clausura transitiva* de $R$
- $R^*$ es la *clausura reflexo-transitiva* de $R$

# Álgebra de relaciones

**Operadores Binarios**:

- $R + S$ es la *unión* de $R$ y $S$
- $R \& S$ es la *intersección* de $R$ y $S$
- $R \cdot S$ es la *composición* de $R$ seguida de $S$
- $R - S$ es la *diferencia* de $R$ respecto de $S$

**Semántica (ejemplos)**:

Si $\mathcal{U}$ es el universo donde se aplican las relaciones, luego

- **iden** $= \{(a, a) \mid a \in \mathcal{U}\}$
- $\breve{R} = \{(b, a) \mid (a, b) \in R\}$
- $R \cdot S = \{(a, b) \in \mathcal{U} \times \mathcal{U} \mid \exists c \in \mathcal{U} : (a, c) \in R \wedge (c, b) \in S\}$

# Álgebra de relaciones

**Algunos Axiomas:**

$R + \textbf{none} = R$

$R \,\&\, \textbf{univ} = R$

$R + S = S + R$

$R \,\&\, S = S \,\&\, R$

$R + R = R$

$R \,\&\, R = R$

$\overline{\textbf{univ}} = \textbf{none}$

$R + \overline{R} = \textbf{univ}$

$R \,\&\, \overline{R} = \textbf{none}$

$\overline{\overline{R}} = R$

$(R + S) \,\&\, T = (R \,\&\, T) + (S \,\&\, T)$

$(R \,\&\, S) + T = (R + T) \,\&\, (S + T)$

$R \cdot \textbf{iden} = \textbf{iden} \cdot R = R$

$(R \cdot S) \cdot T = R \cdot (S \cdot T)$

$(R + S) \cdot T = (R \cdot T) + (S \cdot T)$

$\tilde{}(\tilde{}R) = R$

$\tilde{}(R \cdot S) = \tilde{}S \cdot \tilde{}R$

$\tilde{}(R + S) = \tilde{}R + \tilde{}S$

$\overline{\tilde{}R} = \tilde{}\overline{R}$

$R^+ = R \cdot R^+ + R$

$R^* = (R \cdot R^*) + \textbf{iden}$

# Álgebra de relaciones

**Especificación de propiedades**:

- La relación $R$ es simétrica si

$$R = \tilde{R}$$

- La relación $R$ es transitiva si

$$(R \cdot R)\ \&\ \overline{R} = \textbf{none} \quad \text{o} \quad (R \cdot R) \subseteq R$$

- La relación $R$ tiene dominio $(D \times D)$ si

- La relación $R$ es total si

# Álgebra de relaciones

**Especificación de propiedades**:

- La relación $R$ es simétrica si

$$R = \tilde{R}$$

- La relación $R$ es transitiva si

$$(R \cdot R)\ \&\ \overline{R} = \textbf{none} \quad \text{o} \quad (R$$

$(D \times D)$ es la relación completa en $D$

- La relación $R$ tiene dominio $(D \times D)$ si

$$R\ \&\ \overline{(D \times D)} = \textbf{none} \quad \text{o} \quad R \subseteq (D \times D)$$

- La relación $R$ es total si

$$\textbf{iden} \subseteq R \cdot \tilde{R}$$

# Álgebra de relaciones

**Especificación de propiedades**:

- La relación $R$ es simétrica si

$$R = \tilde{R}$$

- La relación $R$ es transitiva si

$$(R \cdot R)\ \&\ \overline{R} = \textbf{none} \quad \text{o} \quad (R$$

- La relación $R$ tiene dominio $(D \times D)$ si

$$R\ \&\ \overline{(D \times D)} = \textbf{none} \quad \text{o} \quad R \subseteq (D$$

- La relación $R$ es total si

$$\textbf{iden} \subseteq R \cdot \tilde{R}$$

$(D \times D)$ es la relación completa en $D$

Total en la primera coordenada!

## Especificación de propiedades:

- La relación $F$ es una función si

$$\tilde{F} \cdot F \subseteq \textbf{iden} \qquad \text{o} \qquad (\tilde{F} \cdot F) \,\&\, \overline{\textbf{iden}} = \textbf{none}$$

- Si $E$ representa las aristas de un grafo, podemos expresar que el grafo no tiene ciclos mediante

$$E^{+} \,\&\, \textbf{iden} = \textbf{none}$$

- Si $T$ es la relación de transición de un sistema de transiciones etiquetadas, $R$ es una simulación si

$$\tilde{R} \cdot T \subseteq T \cdot \tilde{R}$$