Algoritmos y Estructuras de Datos I

Primer cuatrimestre de 2013

Departamento de Computación - FCEyN - UBA

Especificación - clase 2

Lógica proposicional - tipos básicos

Lógica proposicional - sintaxis

- símbolos

$$\text{true} \ , \ \text{false} \ , \ \bot \ , \ \neg \ , \ \wedge \ , \ \vee \ , \ \rightarrow \ , \ \leftrightarrow \ , \ (\ , \ )$$

- variables proposicionales (infinitas)

$$p \ , \ q \ , \ r \ , \ \ldots$$

- fórmulas
  1. true, false y $\bot$ son fórmulas
  2. cualquier variable proposicional es una fórmula
  3. si A es una fórmula, $\neg A$ es una fórmula
  4. si $A_1, A_2, \ldots, A_n$ son fórmulas, $(A_1 \wedge A_2 \wedge \cdots \wedge A_n)$ es una fórmula
  5. si $A_1, A_2, \ldots, A_n$ son fórmulas, $(A_1 \vee A_2 \vee \cdots \vee A_n)$ es una fórmula
  6. si A y B son fórmulas, $(A \rightarrow B)$ es una fórmula
  7. si A y B son fórmulas, $(A \leftrightarrow B)$ es una fórmula

Ejemplos

¿Cuáles son fórmulas?

- $p \vee q$ no
- $(p \vee q)$ sí
- $p \vee q \rightarrow r$ no
- $(p \vee q) \rightarrow r$ no
- $((p \vee q) \rightarrow r)$ sí
- $(p \rightarrow q \rightarrow r)$ no

Semántica clásica

- 2 valores de verdad posibles
  1. verdadero (1)
  2. falso (0)
- interpretación:
  - true siempre vale 1
  - false siempre vale 0
  - $\neg$ se interpreta como "no", se llama negación
  - $\wedge$ se interpreta como "y", se llama conjunción
  - $\vee$ se interpreta como "o"(no exclusivo), se llama disyunción
  - $\rightarrow$ se interpreta como "si... entonces", se llama implicación
  - $\leftrightarrow$ se interpreta como "si y solo si", se llama doble implicación o equivalencia

Semántica clásica

Conociendo el valor de las variables proposicionales de una fórmula, conocemos el valor de verdad de la fórmula

p	$\neg p$
1	0
0	1

p	q	$(p \wedge q)$
1	1	1
1	0	0
0	1	0
0	0	0

p	q	$(p \vee q)$
1	1	1
1	0	1
0	1	1
0	0	0

p	q	$(p \rightarrow q)$
1	1	1
1	0	0
0	1	1
0	0	1

p	q	$(p \leftrightarrow q)$
1	1	1
1	0	0
0	1	0
0	0	1

Ejemplo: tabla de verdad para $((p \wedge q) \rightarrow r)$

p	q	r	$(p \wedge q)$	$((p \wedge q) \rightarrow r)$
1	1	1	1	1
1	1	0	1	0
1	0	1	0	1
1	0	0	0	1
0	1	1	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	0	1

Semántica trivaluada

- 3 valores de verdad posibles
  1. verdadero (1)
  2. falso (0)
  3. indefinido (−)
- es la que vamos a usar en esta materia
- ¿por qué?
  - queremos especificar problemas que puedan resolverse con un algoritmo
  - puede ser que un algoritmo realice una operación inválida
    - dividir por cero
    - raíz cuadrada de un número negativo
  - necesitamos contemplar esta posibilidad en la especificación
- interpretación:
  - true siempre vale 1
  - false siempre vale 0
  - $\perp$ siempre vale −
  - se extienden las definiciones de $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$

Semántica trivaluada (secuencial)

Se llama secuencial porque

- los términos se evalúan de izquierda a derecha
- la evaluación termina cuando se puede deducir el valor de verdad, aunque el resto esté indefinido

Extendemos la semántica de $\neg$, $\wedge$, $\vee$

p	$\neg p$
1	0
0	1
−	−

p	q	$(p \wedge q)$
1	1	1
1	0	0
0	1	0
0	0	0
1	−	−
0	−	0
−	1	−
−	0	−
−	−	−

p	q	$(p \vee q)$
1	1	1
1	0	1
0	1	1
0	0	0
1	−	1
0	−	−
−	1	−
−	0	−
−	−	−

Semántica trivaluada (secuencial)

Extendemos la semántica de $\rightarrow$, $\leftrightarrow$

p	q	$(p \rightarrow q)$
1	1	1
1	0	0
0	1	1
0	0	1
1	–	–
0	–	1
–	1	–
–	0	–
–	–	–

p	q	$(p \leftrightarrow q)$
1	1	1
1	0	0
0	1	0
0	0	1
1	–	–
0	–	–
–	1	–
–	0	–
–	–	–

Dos conectivos bastan

- $\neg$ y $\vee$
  - $(A \wedge B)$ es $\neg(\neg A \vee \neg B)$
  - $(A \rightarrow B)$ es $(\neg A \vee B)$
  - true es $(A \vee \neg A)$
  - false es $\neg$true
- $\neg$ y $\wedge$
  - $(A \vee B)$ es $\neg(\neg A \wedge \neg B)$
  - $(A \rightarrow B)$ es $\neg(A \wedge \neg B)$
  - false es $(A \wedge \neg A)$
  - true es $\neg$false
- $\neg$ y $\rightarrow$
  - $(A \vee B)$ es $(\neg A \rightarrow B)$
  - $(A \wedge B)$ es $\neg(A \rightarrow \neg B)$
  - true es $(A \rightarrow A)$
  - false es $\neg$true

Tautologías, contradicciones y contingencias

- una fórmula es una tautología si siempre toma el valor 1 para valores definidos de sus variables proposicionales
  Por ejemplo, $((p \wedge q) \rightarrow p)$ es tautología:

p	q	$(p \wedge q)$	$((p \wedge q) \rightarrow p)$
1	1	1	1
1	0	0	1
0	1	0	1
0	0	0	1

- una fórmula es una contradicción si siempre toma el valor 0 para valores definidos de sus variables proposicionales
  Por ejemplo, $(p \wedge \neg p)$ es contradicción:

p	$\neg p$	$(p \wedge \neg p)$
1	0	0
0	1	0

- una fórmula es una contingencia cuando no es ni tautología ni contradicción

Relación de fuerza

Decimos que A es más fuerte que B cuando $(A \rightarrow B)$ es tautología.

También decimos que A fuerza a B o que B es más débil que A.

Por ejemplo,

- ¿$(p \wedge q)$ es más fuerte que p? sí
- ¿$(p \vee q)$ es más fuerte que p? no
- ¿p es más fuerte que $(q \rightarrow p)$? sí
- ¿p es más fuerte que q? no
- ¿p es más fuerte que p? sí
- ¿hay una fórmula más fuerte que todas? sí, por ej. false
- ¿hay una fórmula más débil que todas? sí, por ej. true

Lenguaje de especificación

- hasta ahora vimos lógica proposicional
  - es muy limitada
- nuestro objetivo es especificar (describir problemas)
- vamos a usar un lenguaje más poderoso
  - permite hablar de elementos y sus propiedades
  - es un lenguaje tipado
    - los elementos pertenecen a distintos dominios o conjuntos (enteros, reales, etc.)

Tipos de datos

- conjunto de valores con operaciones
- vamos a empezar viendo tipos básicos
- para hablar de un elemento de un tipo T en nuestro lenguaje, escribimos un término o expresión
  - variable de tipo T
  - constante de tipo T
  - función (operación) aplicada a otros términos (del tipo T o de otro tipo)
- todos los tipos tienen un elemento distinguido: $\perp$ o Indef

Tipo Bool (valor de verdad)

- valores: 1, 0 , $-$
- constantes: true, false, $\perp$ (o Indef)
- conectivos lógicos: $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ con la semántica trivaluada que vimos antes
  - $\neg A$ se puede escribir no(A)
  - $(A \wedge B)$ se puede escribir (A `&&` B)
  - $(A \vee B)$ se puede escribir (A `||` B)
  - $(A \rightarrow B)$ se puede escribir (A `-->` B)
  - $(A \leftrightarrow B)$ se puede escribir (A `<-->` B)
- comparación: $A == B$
  - todos los tipos tienen esta operación (A y B deben ser del mismo tipo T)
  - es de tipo bool
  - es verdadero si el valor de A igual al valor de B (salvo que alguno esté indefinido - ver hoja 20)
  - $A \neq B$ o $A\,!= B$ es equivalente a $\neg(A == B)$
- semántica secuencial

Tipo Int (números enteros)

- sus elementos son los de $\mathbb{Z}$
- constantes: $0 \quad ; \quad 1 \quad ; \quad -1 \quad ; \quad 2 \quad ; \quad -2 \quad ; \quad \ldots$
- operaciones aritméticas
  - $a + b$ (suma)
  - $a - b$ (resta)
  - $a * b$ (multiplicación)
  - a div b (división entera)
  - a mod b (resto de dividir a a por b)
  - a^b o pot(a,b) (potencia)
  - abs(a) (valor absoluto)
- comparaciones (de tipo bool)
  - $a < b$ (menor)
  - $a \leq b$ o a `<=` b (menor o igual)
  - $a > b$ (mayor)
  - $a \geq b$ o a `>=` b (mayor o igual)
- β o beta. Si A es de tipo Bool, se definide como:

$$\beta(A) = \mathrm{beta}(A) = \begin{cases} 1 & \text{si } A \text{ es verdadero} \\ 0 & \text{si } A \text{ es falso} \\ - & \text{si } A \text{ es indefinido} \end{cases}$$

Tipo Float (números reales)

- sus elementos son los de $\mathbb{R}$
- constantes: $0 \quad ; \quad 1 \quad ; \quad -7 \quad ; \quad 81 \quad ; \quad 7{,}4552 \quad ; \quad \pi \ldots$
- operaciones aritméticas
  - las mismas que Int, salvo div y mod
  - a/b (división)
  - $\log_b(a)$ (logaritmo)
  - trigonométricas
- comparaciones (de tipo bool)
  - las mismas que para Int
- conversión a entero
  - $\lfloor a \rfloor$ o $\mathrm{int}(a)$
- todos los términos de tipo Int pueden usarse como términos de tipo Float

Tipo Char (caracteres)

- sus elementos son los las letras, dígitos y símbolos
- constantes: $`a`, `b`, `c`, \ldots, `z`, \ldots, `A`, `B`, `C`, \ldots, `Z`, \ldots, `0`, `1`, `2`, \ldots, `9`$ (en algún orden)
- función ord
  - numera todos los caracteres
  - no importa mucho cuál es el valor de cada uno, pero
    - $\mathrm{ord}(`a`) + 1 == \mathrm{ord}(`b`)$
    - $\mathrm{ord}(`A`) + 1 == \mathrm{ord}(`B`)$
    - $\mathrm{ord}(`1`) + 1 == \mathrm{ord}(`2`)$
- función char
  - es la inversa de ord
- las comparaciones entre caracteres son comparaciones entre sus órdenes
  - $a < b$ es equivalente a $\mathrm{ord}(a) < \mathrm{ord}(b)$
  - lo mismo para $\leq, >, \geq$

Términos

- simples
  - variables del tipo o
  - constantes del tipo
- complejos
  - combinaciones de funciones aplicadas a funciones, constantes y variables

Ejemplos de términos de tipo Int

- $0 + 1$
- $((3 + 4) * 7)^2 - 1$
- $2 * \beta(1 + 1 == 2)$
- $1 + \mathrm{ord}(`A`)$
- con x variable de tipo Int; y de tipo Float; z de tipo Bool
  - $2 * x + 1$
  - $\beta(y^2 > \pi) + x$
  - $(x \bmod 3) * \beta(z)$

Semántica de los términos

- vimos que los términos representan elementos de los tipos
- los términos tienen valor indefinido cuando no se puede hacer alguna operación
  - 1 div 0
  - $(-1)^{1/2}$
- las operaciones son estrictas (salvo los conectivos de bool)
  - si uno de sus argumentos es indefinido, el resultado también está indefinido
  - ejemplos
    - $0 * (-1)^{1/2}$ es indefinido ($*$ es estricto)
    - $0^{1/0}$ es indefinido (pot es estricto)
    - $((1 + 1 == 2) \vee (0 > 1/0))$ es verdadero ($\vee$ no es estricto)
  - las comparaciones con $\perp$ son indefinidas
    - en particular, si x está indefinido, $x == x$ es indefinido (no es verdadero)