Lógica Computacional, 2018-2
Nota 11. Algoritmos de unificación. Regla de resolución y precedimiento de resolución general.*

Noé Salomón Hernández S.

1. Introducción

La diferencia entre resolución en lógica de predicados y resolución en lógica proposicional es la unificación. En lógica proposicional, se les aplicaba la regla de resolución a dos cláusulas si contenían literales complementarias, i.e., la literal positiva era idéntica a la literal negativa, ignorando el símbolo de negación. La misma idea subyace en la resolución en lógica de predicados, excepto que el criterio para clasificar literales complementarias es más relajado. La literal positiva no tiene que ser idéntica a la literal negativa, omitiendo el símbolo de negación; es suficiente con que las dos literales puedan se idénticas al sustituir sus variables.

La unificación es el proceso por el que se determina si dos expresiones pueden ser *unificadas*, i.e., si pueden ser idénticas al sustituir apropiadamente sus variables. Determinar esto es una parte esencial de la resolución en lógica de predicados.

1.1. Ideas previas

Definición 1. Considere los siguientes incisos:

- Un *cláusula* es un conjunto de literales.
- Un *cláusula* es considerada implícitamente como una disyunción de sus literales.
- Una *cláusula unitaria* consiste exactamente de una única cláusula.
- El conjunto vacío de literales es la *cláusula vacía*, denotada como $\square$.
- Una fórmula en *forma clausular* es un conjunto de cláusulas.
- Una fórmula en forma clausular es considerada implícitamente como una conjunción de sus cláusulas.
- La fórmula que es el *conjunto vacío de cláusulas* se denota como $\varnothing$.

*Esta nota se base en el libro de Ben-Ari M., *Mathematical Logic for Computer Science*

Definición 2. Sean:

$$\begin{aligned}\theta &= [x_1 := t_1, \ldots, x_n := t_n] \\ \sigma &= [y_1 := s_1, \ldots, y_k := s_k]\end{aligned}$$

dos sustituciones efectuadas *simultáneamente*, y sean $X = \{x_1, \ldots, x_n\}$ y $Y = \{y_1, \ldots, y_k\}$ los conjuntos de variables que son sustituidas por θ y σ, respectivamente. Así $\theta\sigma$, la *composición de θ y σ*, es la sustitución:

$$\theta\sigma = [x_i := t_i\sigma,\ y_j := s_j] \quad \text{con } x_i \in X,\ x_i \neq t_i\sigma,\ y_j \in Y \text{ y } y_j \notin X$$

2. Unificación

Definición 3. Sea $U = \{A_1, \ldots, A_n\}$ un conjunto de átomos. Un *unificador* θ es una sustitución tal que:

$$A_1\theta = \cdots = A_n\theta.$$

Un *unificador más general (umg)* para U es un unificador μ tal que cualquier unificador θ de U puede ser expresado como:

$$\theta = \mu\lambda$$

para alguna sustitución λ.

3. Algoritmo de unificación por Martelli y Montanari

Si dos átomos son unificables, entonces comparten el mismo símbolo de predicado con la misma paridad. La unificación de átomos se entiende como la unificación de sus argumentos, es decir, la *unificación de un conjunto de términos*. El conjunto de términos a ser unificados será escrito como un conjunto de ecuaciones de términos.

Ejemplo 3.1 La unificación de $\{P(f(x), g(y)), P(f(f(a)), g(z))\}$ se expresa como un conjunto de ecuaciones de términos:

$$f(x) = f(f(a))$$
$$g(y) = g(z)$$

Definición 4. Un conjunto de ecuaciones de términos está en *forma de solución* syss:

- Todas las ecuaciones son de la forma $x_i = t_i$ donde x_i es una variable.
- Cada variable x_i que figura en el lado izquierdo de una ecuación no figura en alguna otra parte del conjunto.

Un conjunto de ecuaciones en forma de solución define la sustitución:

$$[x_1 := t_1, \ldots, x_n := t_n]$$

Algoritmo (Algoritmo de unificación de Martelli y Montanari)
Input: Un conjunto de ecuaciones de términos.
Output: Un conjunto de ecuaciones en forma de solución o reportar *no unificable*.

Ejecutar las siguientes transformaciones sobre el conjunto de ecuaciones mientras alguna de ellas sea aplicable:

1. Transformar $t = x$, donde t no es una variable, en $x = t$.

2. Eliminar la ecuación $x = x$.

3. Sea $t = s$ una ecuación donde t, s no son variables,

   - Si el símbolo de función más externo de t y s no es el mismo, el algoritmo termina reportando *no unificable*.
   - Si el símbolo de función más externo de t y s sí es el mismo, pero difiere en aridad, el algoritmo termina reportando *no unificable*.
   - De otro modo, reemplazar la ecuación $f(t_1, \ldots, t_k) = f(s_1, \ldots, s_k)$ por las k ecuaciones $t_1 = s_1, \ldots, t_k = s_k$.

4. Sea $x = t$ una ecuación tal que x figura en otro lugar dentro del conjunto.

   - Si x figura en t, y el término t no es la variable x misma, el algoritmo termina reportando *no unificable*.
   - De otro modo, transformar el conjunto al reemplazar todas las presencias de x por t en las otras ecuaciones donde x figura.

Los algoritmos de unificación pueden ser bastante ineficientes por la verificación de la condición descrita en la regla 4, llamada *verificación de presencias*.

En la aplicación de la unificación en la programación lógica, la verificación de presencias se ignora y se toma el riesgo de una sustitución ilegal.

Ejemplo 3.1. Considere el siguiente conjunto con dos ecuaciones de términos:

$$\begin{aligned} g(y) &= x, \\ f(x, h(x), y) &= f(g(z), w, z). \end{aligned}$$

Aplicamos la regla 1 a la primera ecuación y la regla 3 a la segunda ecuación:

$$\begin{aligned} x &= g(y), \\ x &= g(z), \\ h(x) &= w, \\ y &= z. \end{aligned}$$

Aplicamos la regla 4 a la segunda ecuación reemplazando la presencias de x por $g(z)$ en las demás ecuaciones:

$$\begin{aligned} g(z) &= g(y), \\ x &= g(z), \\ h(g(z)) &= w, \\ y &= z. \end{aligned}$$

Aplicamos la regla 3 a la segunda ecuación:

$$\begin{aligned} z &= y, \\ x &= g(z), \\ h(g(z)) &= w, \\ y &= z. \end{aligned}$$

Aplicamos la regla 4 a la última ecuación reemplazando y por z en la primera ecuación; esto resulta en la ecuación $z = z$ la cual eliminamos usando la regla 2:

$$\begin{aligned} x &= g(z), \\ h(g(z)) &= w, \\ y &= z. \end{aligned}$$

Finalmente, transformamos la segunda ecuación mediante la regla 1:

$$\begin{aligned} x &= g(z), \\ w &= h(g(z)), \\ y &= z. \end{aligned}$$

Así terminamos exitosamente el algoritmo, y afirmamos que

$$\mu = [x := g(z), w := h(g(z)), y := z]$$

es **un** unificador más general del conjunto original de ecuaciones. Podemos verificar también que el unificador:

$$\theta = [x := g(h(b)), w := h(g(h(b))), y := h(b), z := h(b)]$$

puede expresarse como $\theta = \mu\,[z := h(b)]$.

4. Algoritmo de unificación de Robinson

Definición 5. Sean A y A' dos átomos con el mismo símbolo de predicado y misma aridad. Consideremos a cada uno de dichos átomos como una secuencia de símbolos. Sea k la posición más a la izquierda en la que las secuencias difieren. El par de términos $\{t, t'\}$, que comienzan en la posición k de A y A', es el *conjunto de desacuerdo* de los dos átomos.

Algoritmo (Algoritmo de unificación de Robinson)
Input: Dos átomos A y A' con el mismo símbolo de predicado y misma aridad.
Output: Un unificador más general para A y A' o reportar *no unificable*.

Inicializar el algoritmo al definir $A_0 = A$ y $A'_0 = A'$. Ejecutar sucesivamente la siguiente acción:

- Sea $\{t, t'\}$ el conjunto de desacuerdo de A_i y A'_i. Si un término es una variable x_{i+1} y el otro es un término t_{i+1} tal que x_{i+1} no figura en t_{i+1}, entonces definimos $\sigma_{i+1} = [x_{i+1} := t_{i+1}]$, y generamos los átomos $A_{i+1} = A_i\sigma_{i+1}$ y $A'_{i+1} = A'_i\sigma_{i+1}$ para una nueva iteración del algoritmo.

Si no es posible ejecutar esta acción (debido a que ambos elementos del conjunto de desacuerdo, $\{t, t'\}$, no son variables; o debido a que un término, digamos t, es una variable y ésta figura en el otro término, t'), los átomos no son unificables. Si después de ejecutar esta acción $A_n = A'_n$, entonces A, A' son unificables y el umg es $\mu = \sigma_1 \cdots \sigma_n$.

Ejemplo 4.1. Considere el par de átomos:

$$A = P(g(y), f(x, h(x), y)), \quad A' = P(x, f(g(z), w, z)).$$

Comenzamos definiendo $A_0 = A$ y $A'_0 = A'$. El conjunto de desacuerdo inicial es $\{x, g(y)\}$. Un término es la variable x que no está presente en el otro término, así $\sigma_1 = [x := g(y)]$, y:

$$A_0\sigma_1 = A_1 = P(g(y), f(g(y), h(g(y)), y)),$$

$$A'_0\sigma_1 = A'_1 = P(g(y), f(g(z), w, z)).$$

El conjunto de desacuerdo de A_1 y A'_1 es $\{y, z\}$ así $\sigma_2 = [y := z]$, y:

$$A_1\sigma_2 = A_2 = P(g(z), f(g(z), h(g(z)), z)),$$

$$A'_1\sigma_2 = A'_2 = P(g(z), f(g(z), w, z)).$$

El conjunto de desacuerdo de A_2 y A'_2 es $\{w, h(g(z))\}$ así $\sigma_3 = [w := h(g(z))]$, y:

$$A_2\sigma_3 = A_3 = P(g(z), f(g(z), h(g(z)), z)),$$

$$A'_2\sigma_3 = A'_3 = P(g(z), f(g(z), h(g(z)), z)).$$

Como $A_3 = A'_3$, los átomos son unificables y un umg es:

$$\mu = \sigma_1\sigma_2\sigma_3 = [x := g(z), y := z, w := h(g(z))].$$

5. Resolución general

Definición 6. Sea $L = \{l_1, \ldots, l_n\}$ un conjunto de literales. Entonces $L^c = \{l_1^c, \ldots, l_n^c\}$, es decir, L^c es el conjunto de literales complementarias de las que se encuentran en L.

Definición 7. (**Regla de resolución general**) Sean C_1 y C_2 cláusulas sin variables en común. Sean $L_1 = \{l_1^1, \ldots, l_{n_1}^1\} \subseteq C_1$ y $L_2 = \{l_1^2, \ldots, l_{n_2}^2\} \subseteq C_2$ dos subconjuntos de literales tales que L_1 y L_2^c se unifican por un umg σ. C_1 y C_2 se conocen como cláusulas de colisión y colisionan en el conjunto de literales L_1 y L_2. C, el resolvente de C_1 y C_2, es la cláusula:

$$Res(C_1, C_2) = (C_1\sigma - L_1\sigma) \cup (C_2\sigma - L_2\sigma).$$

Ejemplo 5.1. Sean $C_1 = \{P(f(x), g(y)), Q(x, y)\}$ y $C_2 = \{\neg P(f(f(a)), g(z)), Q(f(a), z)\}$ dos cláusulas, un umg para $L_1 = \{P(f(x), g(y))\}$ y $L_2^c = \{P(f(f(a)), g(z))\}$ es $[x := f(a), y := z]$. El resolvente para las cláusulas C_1 y C_2 es

$$\{Q(f(a), z), Q(f(a), z)\} = \{Q(f(a), z)\}$$

Las cláusulas con *conjuntos* de literales, así que cuando tomamos la unión de las cláusulas en la regla de resolución, literales idénticas se colapsarán; esto se llama *factorización*.

La regla de resolución general requiere que las cláusulas no tengan variables en común. Esto se logra mediante un proceso que se conoce como *estandarización por partes*: renombramos todas las variables en una de las cláusulas antes de que ser usada por la regla de resolución. Todas las variables en una cláusulas están implícitamente cuantificadas universalmente así que el renombre no cambia su satisfacibilidad.

Ejemplo 5.2. Sean $C_1 = \{P(x), P(y)\}$ y $C_2 = \{\neg P(x), \neg P(y)\}$ dos cláusulas. Aplicando estandarización por partes tenemos que $C_2^{'} = \{\neg P(x'), \neg P(y')\}$. Sea $L_1 = \{P(x), P(y)\}$ y $L_2^c = \{P(x'), P(y')\}$; estos conjuntos tienen un umg $\sigma = [y := x,\ x' := x,\ y' := x]$. La regla de resolución opera del siguiente modo:

$$\begin{array}{rcl} Res(C_1, C_2) & = & (C_1\sigma - L_1\sigma) \cup (C_2^{'}\sigma - L_2\sigma) \\ & = & (\{p(x)\} - \{p(x)\}) \cup (\{\neg p(x)\} - \{\neg p(x)\}) \\ & = & \Box \end{array}$$

Algoritmo (Procedimiento de Resolución General)
Input: Un conjunto de cláusulas S.
Output: Si el algoritmo termina, reporta que el conjunto de cláusulas es *satisfacible* o *insatisfacible*.

Sea $S_0 = S$. *Escoga* dos cláusulas de colisión $C_1, C_2 \in S_i$ y sea $C = Res(C_1, C_2)$. Si $C = \Box$, el algoritmo termina y reporta que S es *insatisfacible*. De otro modo, construir $S_{i+1} = S_i \cup \{C\}$. Si $S_{i+1} = S_i$ para todos los posibles pares de cláusulas de colisión, el algoritmo termina y reporta S es *satisfacible*.

Aunque un conjunto de cláusulas insatisfacible producirá eventualmente $\Box$ con una adecuada ejecución sistemática del procedimiento, la existencia de modelos infinitos significa que el procedimiento de resolución sobre un conjunto de cláusulas satisfacible puede **nunca** terminar, así que el prodecimiento de resolución general no es un procedimiento de decisión.

Ejemplo 5.3. Las líneas de la 1 a la 7 contienen un conjunto de cláusulas inicial. La refutación por resolución a partir de la línea 8 muestra que el conjunto de cláusulas inicial es insatisfacible.

Cada linea contiene el resolvente, el unificador más general y los números con las cláusulas padre.

1.	$\{\neg P(x), Q(x), R(x, f(x))\}$		*premisa*
2.	$\{\neg P(x), Q(x), O(f(x))\}$		*premisa*
3.	$\{M(a)\}$		*premisa*
4.	$\{P(a)\}$		*premisa*
5.	$\{\neg R(a, y), M(y)\}$		*premisa*
6.	$\{\neg M(x), \neg Q(x)\}$		*premisa*
7.	$\{\neg M(x), \neg O(x)\}$		*premisa*
8.	$\{\neg Q(a)\}$	$[x := a]$	$3, 6$
9.	$\{Q(a), O(f(a))\}$	$[x := a]$	$2, 4$
10.	$\{O(f(a))\}$		$8, 9$
11.	$\{Q(a), R(a, f(a))\}$	$[x := a]$	$1, 4$
12.	$\{R(a, f(a))\}$		$8, 11$
13.	$\{M(f(a))\}$	$[y := f(a)]$	$5, 12$
14.	$\{\neg O(f(a))\}$	$[x := f(a)]$	$7, 13$
15	$\square$		$10, 14$

6. El método de resolución general el correcto y completo

Teorema 1 (La resolución es correcta)**.** *Sea S un conjunto de cláusulas. Si al aplicar el procedimiento de resolución se obtiene la cláusula vacía $\square$, entonces S es insatisfacible.*

Teorema 2 (La resolución es completa)**.** *Si un conjunto de cláusulas es insatisfacible, la cláusula vacía $\square$ puede obtenerse mediante el procedimiento de resolución.*