Lógica Computacional 2016-2, nota de clase 9
Formas Normales en Lógica de Predicados

Favio Ezequiel Miranda Perea   Araceli Liliana Reyes Cabello
Lourdes Del Carmen González Huesca

Facultad de Ciencias UNAM
7 de abril de 2016

En esta nota desarrollaremos diversas formas normales para la lógica de predicados, las cuales son transformaciones sintácticas de una fórmula que conservan ciertas propiedades semánticas. Las fórmulas en forma normal, si bien pueden ser sintácticamente más complejas, ayudan a semiautomatizar en ciertos casos el proceso para tratar de verificar si un argumento es correcto. Además son de gran importancia en programación lógica.

Antes de construir formas normales se aplica un proceso de simplificación a las fórmulas llamado *rectificación*, el cual permite simplificar la transformación a la forma normal de interés.

Definición 1 *Una fórmula φ esta rectificada si y sólo si se cumplen las siguientes condiciones:*

- *φ no tiene presencias libres y ligadas de una misma variable, es decir, $FV(\varphi) \cap BV(\varphi) = \varnothing$.*
- *φ no tiene cuantificadores de la misma variable con alcances ajenos.*
- *φ no tiene cuantificadores multiples de la misma variable, es decir, subfórmulas de la forma*

$$\forall x \forall x \psi, \ \exists x \exists x \psi, \ \exists x \forall x \psi, \ \forall x \exists x \psi.$$

- *φ no tiene cuantificadores vácuos (tontos), es decir, cuantificadores de la forma $\forall x \psi$ o $\exists x \psi$ donde $x \notin FV(\psi)$.*

Nuestro primer objetivo es mostrar que cualquier fórmula es rectificable para lo cual nos serviremos del siguiente

Lema 1 *Se cumplen las siguientes equivalencias lógicas.*

1. *Eliminación de cuantificadores múltiples.*

   a) $\forall x \forall x \varphi \equiv \forall x \varphi$.
   b) $\exists x \exists x \varphi \equiv \exists x \varphi$.
   c) $\exists x \forall x \varphi \equiv \forall x \varphi$.
   d) $\forall x \exists x \varphi \equiv \exists x \varphi$.

2. Renombre de variables. Si y no figura libre en φ, es decir, $y \notin FV(\varphi)$ entonces:

a) $\forall x \varphi \equiv \forall y(\varphi[x := y])$.

b) $\exists x \varphi \equiv \exists y(\varphi[x := y])$.

3. Eliminación de cuantificadores vácuos. Si $x \notin FV(\varphi)$ entonces:

a) $\forall x \varphi \equiv \varphi$.

b) $\exists x \varphi \equiv \varphi$.

Demostración. Se deja como ejercicio. $\dashv$

Una consecuencia inmediata de los lemas anteriores es la siguiente proposición.

Proposición 1 *Siempre es posible transformar mediante un algoritmo una fórmula dada φ en una fórmula $rec(\varphi)$ de manera que $rec(\varphi)$ está rectificada y $rec(\varphi) \equiv \varphi$.*

Demostración. Sea φ una fórmula. Si $FV(\varphi) \cap BV(\varphi) \neq \varnothing$ podemos aplicar el lema 1(parte 2) para renombrar las presencias ligadas de todas las variables que figuraban libres y ligadas en φ, análogamente si existian en φ cuantificadores con alcances ajenos de la misma variable. Los cuantificadores múltiples se eliminan mediante el lema 1(parte 1) y los cuantificadores vácuos con el lema 1(parte 3). Cualquier fórmula obtenida al terminar estos procesos se denota $rec(\varphi)$, obsérvese que no es única. $\dashv$

Nuestro objetivo principal es transformar una fórmula de la dada de la lógica de predicados en una fórmula esencialmente proposicional que sea en cierto sentido equivalente a la fórmula original. Con las siguientes transformaciones esto se logra parcialmente.

1. Forma Normal Negativa

El objetivo de esta forma normal es obtener una fórmula equivalente a una fórmula dada de manera que las negaciones solo afecten a fórmulas atómicas, es decir a predicados.

Definición 2 *Una fórmula φ está en forma normal negativa si y sólo si se cumplen las siguientes condiciones:*

1. φ no contiene ni equivalencias ni implicaciones.

2. las negaciones que figuran en φ afectan sólo a fórmulas atómicas.

La transformación a forma normal negativa es prácticamente igual que en el caso de la lógica proposicional, basta considerar adicionalmente las leyes de negación para los cuantificadores.

Proposición 2 (Leyes de la Negación) *Se cumplen las siguientes equivalencias lógicas.*

- $\neg\forall x\varphi \equiv \exists x\neg\varphi$
- $\neg\exists x\varphi \equiv \forall x\neg\varphi$

Demostración. Ejercicio $\dashv$

Proposición 3 *Sea φ una fórmula. Podemos encontrar de manera algorítmica una fórmula ψ logicamente equivalente a φ y tal que ψ está en forma normal negativa. En tal caso ψ se denota con $fnn(\psi)$.*

Demostración. Dada φ, aplicarle exhaustivamente las leyes de negación de manera adecuada. La fórmula ψ obtenida es $fnn(\varphi)$. $\dashv$

Ejemplo 1.1 Sea $\varphi = \forall x(Px \vee \neg\exists y(Qy \wedge Rxy)) \vee \neg(Py \wedge \neg\forall xPx)$.
La forma normal negativa de $\neg\exists y(Qy \wedge Rxy)$ se obtiene así:

$$\neg\exists y(Qy \wedge Rxy) \equiv \forall y\neg(Qy \wedge Rxy) \equiv \forall y(\neg Qy \vee \neg Rxy)$$

La forma normal negativa de $\neg(Py \wedge \neg\forall xPx)$ se obtiene así:

$$\neg(Py \wedge \neg\forall xPx) \equiv \neg Py \vee \neg\neg\forall xPx \equiv \neg Py \vee \forall xPx$$

Por lo tanto tenemos que $fnn(\varphi) = \forall x(Px \vee \ \forall y(\neg Qy \vee \neg Rxy)) \vee (\neg Py \vee \forall xPx)$.

2. Forma Normal Prenex

El primer paso para eliminar los cuantificadores de una fórmula consiste en *factorizarlos* de manera que todos queden juntos al principio de la fórmula, esto se logra con la llamada forma normal prenex.
El siguiente lema describe las propiedades básicas de factorización o prenexación de cuantificadores.

Lema 2 *Se cumplen las siguientes equivalencias lógicas.*

1. *Si x no figura libre en φ entonces:*
   a) $\varphi \wedge \forall x\psi \equiv \forall x(\varphi \wedge \psi)$.
   b) $\varphi \wedge \exists x\psi \equiv \exists x(\varphi \wedge \psi)$.
   c) $\varphi \vee \forall x\psi \equiv \forall x(\varphi \vee \psi)$.
   d) $\varphi \vee \exists x\psi \equiv \exists x(\varphi \vee \psi)$.
   e) $\varphi \to \forall x\psi \equiv \forall x(\varphi \to \psi)$.
   f) $\varphi \to \exists x\psi \equiv \exists x(\varphi \to \psi)$.
2. *Si x no figura libre en ψ entonces:*

a) $\forall x\varphi \to \psi \equiv \exists x(\varphi \to \psi)$.
b) $\exists x\varphi \to \psi \equiv \forall x(\varphi \to \psi)$.

Demostración. Se deja como ejercicio. $\dashv$

Obsérvese que no se dieron equivalencias lógicas para factorizar un cuantificador en una equivalencia, como $\varphi \leftrightarrow \forall x\psi$. De manera que primero deben eliminarse las equivalencias mediante la implicación y la conjunción y luego prenexar.

De mayor importancia es tener en cuenta que al prenexar un cuantificador en el antecedente de una implicación éste se intercambia por el cuantificador contrario.

Definición 3 *Una fórmula φ está en forma normal prenex si y sólo si φ es de la forma*

$$Q_1x_1 \dots Q_nx_n\psi$$

donde ψ es una fórmula sin cuantificadores y $Q_i \in \{\forall, \exists\}$ para toda $1 \leq i \leq n$. En tal caso la cadena de cuantificadores $Q_1x_1 \dots Q_nx_n$ se conoce como el prefijo *de φ mientras que ψ es la* matriz *de φ.*

Proposición 4 *Sea φ una fórmula. Se puede construir mediante un algoritmo una fórmula ψ en forma normal prenex tal que $\varphi \equiv \psi$. A ψ se le denota con $fnp(\varphi)$. Obsérvese que $fnp(\varphi)$ no es única.*

Demostración. Dada φ eliminar las bicondicionales[1] obteniendo una fórmula φ'. Aplicarle el lema 2 a la fórmula $rec(\varphi')$ hasta factorizar todos sus cuantificadores. La fórmula obtenida es $fnp(\varphi)$. $\dashv$

Ejemplo 2.1 Sea φ la misma fórmula del ejemplo 1.1.
Tenemos $rec(\varphi) = \forall x(Px \lor \neg\exists v(Qv \land Rxv)) \lor \neg(Py \land \neg\forall zPz)$.
La forma normal prenex de $\neg\exists v(Qv \land Rxv)$ es $\forall v\neg(Qv \land Rxv)$.
La forma normal prenex de $\forall x(Px \lor \forall v\neg(Qv \land Rxv))$ es $\forall x\forall v(Px \lor \neg(Qv \land Rxv))$.
La forma normal prenex de $\neg\forall zPz$ es $\exists z\neg Pz$.
La forma normal prenex de $Py \land \exists z\neg Pz$ es $\exists z(Py \land \neg Pz)$.
La forma normal prenex de $\neg\exists z(Py \land \neg Pz)$ es $\forall z\neg(Py \land \neg Pz)$.
Por lo tanto la forma normal prenex de φ es

$$fnp(\varphi) = \forall x\forall v\forall z(Px \lor \neg(Qv \land Rxv) \lor \neg(Py \land \neg Pz)).$$

Ejemplo 2.2 Sea $\varphi = \forall xPx \to \exists y\forall zRyz$. Entonces

$$\begin{array}{ll} \forall xPx \to \exists y\forall zRyz & \equiv \\ \exists y(\forall xPx \to \forall zRyz) & \equiv \\ \exists y\forall z(\forall xPx \to Ryz) & \equiv \\ \exists y\forall z\exists x(Px \to Ryz). & \end{array}$$

$\therefore \quad fnp(\varphi) = \exists y\forall z\exists x(Px \to Ryz)$.

[1]De la manera más conveniente en cada caso particular, ya sea usando $\varphi \leftrightarrow \psi \equiv (\varphi \to \psi) \land (\psi \to \varphi)$ o bien $\varphi \leftrightarrow \psi \equiv (\varphi \land \psi) \lor (\neg\varphi \land \neg\psi)$

3. Forma Normal de Skolem

Ahora nuestro interés se centra en los cuantificadores existenciales. Buscamos un método para eliminarlos de una fórmula, de manera que se preserve la satisfacibilidad. Para esto utilizaremos el proceso de *skolemización*[2], el cual consiste en substituir las fórmulas existenciales por testigos nuevos de las cuantificaciones existenciales de la fórmula, de la siguiente manera:

$$\begin{array}{lll} \exists x\varphi & \text{se substituye por} & \varphi[x := c] \\ \forall x_1 \ldots \forall x_n \exists y \varphi & \text{se substituye por} & \forall x_1 \ldots \forall x_n \varphi[y := g x_1 \ldots x_n] \end{array}$$

aquí, c es un símbolo de constante **nuevo**, llamado *constante de Skolem* y $g^{(n)}$ es un símbolo de función **nuevo** llamado *función de Skolem*. Con $Sko(\varphi)$ denotamos a la skolemización de φ que se obtiene al eliminar mediante este proceso, de izquierda a derecha, todos los cuantificadores existenciales de la fórmula φ.

Ejemplo 3.1

φ	$sko(\varphi)$
$\exists y Ry$	Rc
$\forall x \exists y Pxy$	$\forall x Pxgx$
$\forall x \forall z \exists y Rxgyfz$	$\forall x \forall z Rxghxzfz$

Hasta ahora hemos visto que una fórmula φ es lógicamente equivalente a cualquiera de sus formas normales, situación muy conveniente pues si hallamos un modelo de una de las formas normales, éste mismo nos servira como modelo de φ. Desafortunadamente no sucede lo mismo con la skolemización, sin embargo tenemos una relación más debil entre φ y $sko(\varphi)$, la cual resultará suficiente para nuestros propósitos.

Definición 4 *Sean φ, ψ dos fórmulas, decimos que φ, ψ son equisatisfacibles cuando se cumple que: φ es satisfacible si y sólo si ψ es satisfacible. Esta propiedad sera denotada con $\varphi \sim_{sat} \psi$.*

Es claro que $\varphi \equiv \psi$ implica que $\varphi \sim_{sat} \psi$, mas no al reves.

Ejemplo 3.2 Claramente $\forall x Pxa \sim_{sat} \forall x Pax$ pero si interpretamos a como 0 y P como $\leq$ sobre $\mathbb{N}$ observamos que $\forall x Pxa \not\equiv \forall x Pax$.

Ejemplo 3.3 $\exists x Qx \sim_{sat} Qc$, este ejemplo muestra que una fórmula equisatisfacible a la fórmula existencial dada, resulta mucho más fácil de analizar.

Proposición 5 *Si φ es una fórmula entonces $\varphi \sim_{sat} Sko(\varphi)$*

[2]El nombre es debido a su creador, Thoralf Skolem (1887-1963).

Demostración. Ejercicio. $\dashv$

Finalmente podemos construir la forma normal de Skolem de cualquier fórmula.

Definición 5 *Una enunciado de la forma*

$$\forall x_1 \ldots \forall x_n \varphi,$$

donde φ no tiene cuantificadores y además φ está en forma normal conjuntiva es una fórmula en forma normal de Skolem.

Obsérvese que en la definición anterior se pide que la fórmula sea un enunciado, es decir, que no tenga variables libres. Si de entrada existiera alguna variable libre se cuantifica universalmente, situación que no cambia en absoluto los resultados pues el buscar modelos para fórmulas con variables libres equivale a buscar modelos para sus cerraduras universales.

Proposición 6 *Para cada fórmula φ podemos construir efectivamente una fórmula ψ en forma normal de Skolem de manera que $\varphi \sim_{sat} \psi$. En tal caso ψ se denota con $fns(\varphi)$.*

Demostración. Es consecuencia inmediata de los resultados anteriores de esta nota. $\dashv$

Ejemplo 3.4 Sea $\varphi = \forall x \exists y (\exists z \forall w Rxyzw \to \exists v Pv)$. Primero obtenemos la forma normal prenex de φ:

$$\varphi \equiv \forall x \exists y \exists v (\exists z \forall w Rxyzw \to Pv) \equiv \forall x \exists y \exists v \forall z (\forall w Rxyzw \to Pv) \equiv$$

$$\forall x \exists y \exists v \forall z \exists w (Rxyzw \to Pv) = fnp(\varphi)$$

Ahora aplicamos el proceso de skolemización a $fnp(\varphi)$:

1. Eliminamos el primer existencial $\exists y$ mediante la función de skolem $f^{(1)}$, obteniendo
$$\forall x \exists v \forall z \exists w (Rxfxzw \to Pv).$$
2. Eliminamos el segundo existencial $\exists v$ mediante la función de skolem $g^{(1)}$, obteniendo
$$\forall x \forall z \exists w (Rxfxzw \to Pgx).$$
3. Eliminamos el tercer existencial $\exists w$ mediante la función de skolem $h^{(2)}$, obteniendo
$$\forall x \forall z (Rxfxzhxz \to Pgx).$$

$\therefore \quad sko(\varphi) = \forall x \forall z (Rxfxzhxz \to Pgx).$

Finalmente para obtener $fns(\varphi)$ basta poner la matriz de $sko(\varphi)$ en forma normal conjuntiva[3].

$$fnc(Rxfxzhxz \to Pgx) \equiv \neg Rxfxzhxz \vee Pgx.$$

Por lo tanto la forma normal de Skolem buscada es

$$fns(\varphi) = \forall x \forall z (\neg Rxfxzhxz \vee Pgx)$$

[3] La transformación a forma normal conjuntiva es exactamente igual que en el caso de la lógica de proposiciones sólo que en lugar de variables proposicionales se manejan predicados.

4. Forma Clausular

Finalmente llegamos a la forma normal necesaria para el método de resolución binaria y para la programación lógica, la llamada forma clausular.

Definición 6 *Sean φ una fórmula y $fns(\varphi) = \forall x_1 \ldots \forall x_n \psi$ la forma normal de Skolem de φ. Si ψ es $\mathcal{C}_1 \wedge \ldots \wedge \mathcal{C}_n$ entonces la forma clausular de φ, denotada $Cl(\varphi)$ es la secuencia de cláusulas*

$$Cl(\varphi) = \mathcal{C}_1, \ldots, \mathcal{C}_n$$

donde sin perder generalidad *las variables en cualesquiera dos cláusulas son distintas. Es decir, cualesquiera dos cláusulas tienen variables ajenas.*

De manera que la forma clausular de una fórmula φ se obtiene de la matriz de su forma normal de Skolem.
Observemos que $Cl(\varphi)$ es una secuencia (conjunción) de cláusulas, libre de cuantificadores; además es inmediato que existe un proceso efectivo para construir $Cl(\varphi)$ y que $Cl(\varphi) \sim_{sat} \varphi$. Con esto hemos logrado el objetivo planteado, cada fórmula φ puede transformarse en una fórmula esencialmente proposicional $Cl(\varphi)$, satisfaciblemente equivalente a φ.

Ejemplo 4.1 Sea φ la fórmula del ejemplo 3.4. La forma cláusular de φ es

$$Cl(\varphi) = \neg Rxfxzhxz \vee Pgx$$

Esto concluye nuestra exposición de formas normales, por medio de la cual ahora podemos transformar cualquier fórmula en su forma clausular, la cual es una fórmula en forma normal conjuntiva sin cuantificadores y equisatisfacible a la fórmula original.

5. Algoritmo de Conversión a la Forma Clausular

La conversión a forma clausular de cualquier fórmula φ se obtiene al aplicar todas las formas normales anteriores. Damos aquí un algoritmo que se sirve además de ciertas heurísticas para facilitar el proceso.

1. Rectificar φ
2. Transformar a la forma normal negativa
3. Prenexar obteniendo una fórmula de la forma $Qz_1 \ldots Qz_m \chi$
4. Skolemizar obteniendo una fórmula de la forma $\forall x_1 \ldots \forall x_n \psi$
5. Transformar ψ (la matriz de la skolemización anterior) a la forma normal conjuntiva obteniendo $fnc(\psi) = \mathcal{C}_1 \wedge \ldots \wedge \mathcal{C}_k$
6. Renombrar las variables de cada cláusula $\mathcal{C}_i$ obteniendo una clásula $\mathcal{C}'_i$ de manera que las variables en cada una de estas sean distintas.

7. La forma clausular de φ es
$$Cl(\varphi) = \mathcal{C}'_1, \ldots, \mathcal{C}'_k$$

De esta manera concluimos que cualquier fórmula de la lógica de predicados puede transformarse en un conjunto de fórmulas de la llama lógica clausular.

Nuestro siguiente tema versa sobre otro método sintáctico para verificar la correctud de un argumento dentro de la lógica clausular, la *resolución binaria*, método que es también el fundamento principal del paradigma de programación lógica.