Revista Integración
Escuela de Matemáticas
Universidad Industrial de Santander
Vol. 32, No. 2, 2014, pág. 199–210

Una coinstitución para la lógica de comportamiento abstracto

Jaime Andrés Castaño Perea*, Guillermo Ortiz Rico,

Universidad del Valle, Departamento de Matemáticas, Cali, Colombia.

Resumen. Recientemente, la especificación de un problema en ciencias de la computación –un paso intermedio entre el problema dado y su aplicación como un sistema de *software* que garantiza su solución– utiliza el álgebra universal y la teoría de coálgebras para su descripción. Esta etapa incluye componentes sintácticas y semánticas, que tienen como resultado un sistema lógico. En [3], se propone la lógica ecuacional multitipada para la especificación de problemas. Dualmente, en [9] se estudia una lógica de comportamiento abstracto, la cual modela procesos y comportamiento de sistemas coalgebraicos. En ambas lógicas los componentes sintáctico y semántico son conectados por medio de una relación de satisfacción, caracterizada por el siguiente principio: la verdad se preserva bajo transformaciones del lenguaje. En un marco general y moderno, hoy contamos con las instituciones en la especificación algebraica y coinstituciones en la especificación coalgebraica. El propósito del presente artículo es estudiar un caso particular de la lógica de comportamiento abstracto presentada en [9], en donde las coálgebras las restringimos a funtores polinomiales. Identificamos la respectiva coinstitución coalgebraica, detallando sus componentes y explícitamente presentaremos la relación de satisfacción como un resultado final.

Keywords: Especificación, categoría, álgebra, coálgebra, relación de satisfacción, institución, coinstitución.
MSC2010: 18A05, 18C10, 03B70, 68Q65.

A coinstitution for abstract behavioral logic

Abstract. Recently, the specification of a problem in computer sciences –an intermediate step between the given problem and its implementation as a software system that guarantees its solution– uses universal algebra and coalgebra theories for its description. This stage includes a syntactic and a semantic component, having a logic system as result. In [3], the case of many-sorted equational logic is studied for the purpose of specification problems. Dually, in [9] an abstract behavioral logic, which models processes and

* E-mail: jaime.castano@correounivalle.edu.co
Recibido: 11 de diciembre de 2013, Aceptado: 23 de agosto de 2014.
Para citar este artículo: J.A. Castaño Perea, G. Ortiz Rico, Una coinstitución para la lógica de comportamiento abstracto, *Rev. Integr. Temas Mat.* 32 (2014), no. 2, 199-210.

coalgebraic systems behavior is studied. In both logics, the syntactic and semantic components are connected via a satisfaction relation, characterized by the following principle: the truth of formulas is invariant under language translations. In a general and modern framework, we use the institutions in algebraic specification and coinstitutions in coalgebraic specification. We research a particular case of behavioral abstract logic presented in [9], in which coalgebras are restricted to polinomial functors. We identify the respective algebraic coinstitution, detail all its components, and explicitly present the satisfaction relation as the final result.
Palabras clave: Specification, categories, algebra, coalgebra, satisfaction relation, institution, coinstitution.

1. Introducción

Actualmente, uno de los problemas fundamentales en el área de ciencias de la computación es el de verificar si un programa funciona correctamente, es decir, si funciona como los usuarios esperan de él. Para garantizar tal correctitud, describir sin ambigüedades lo que hace el programa se convierte en un objetivo principal, siendo los fundamentos formales una herramienta central para garantizar que la especificación (descripción) de un programa posea una única lectura. Un programa es visto como una sucesión de instrucciones que se especifican en un lenguaje y que gozan de ciertas relaciones, procesos y estructuras entre ellas. En el ejercicio de abstracción para modelar programas se ha incurrido en el uso de teorías que permitan incorporar todas las instrucciones empleando la teoría de álgebra y coálgebra universal, en donde esta última usa ampliamente el lenguaje de teoría de categorías para su descripción. El tipo de requerimientos que deben probarse como correctos ha obligado a ir cada vez más lejos en las especificaciones, al punto de modelar la dinámica funcional de un sistema de *software*, es decir, el comportamiento reactivo de este con base en los sucesivos cambios de estados internos. La especificación está compuesta de un conjunto de fórmulas que son construidas a partir de un lenguaje, constituyendo los axiomas que modelan el comportamiento funcional o reactivo del sistema. Esto describe por lo general un sistema lógico que permite precisar lo que el programa debe hacer, dejando la especificación como un procedimiento de naturaleza sintáctica.

Desde la década de los 70 se han venido incorporando técnicas algebraicas en la especificación de programas. Las álgebras gozan de la ventaja de haber sido estudiadas ampliamente y cuentan con un arsenal de herramientas para su comprensión. Estas permiten expresar matemáticamente la obtención de objetos a partir de otros a través de operaciones; y en muchos casos son las semánticas de ciertas lógicas. Estas dos características son fundamentales para una teoría que pretenda modelar un programa con fines de estudiar su correctitud. Es así que la teoría de modelos clásica proporciona la especificación algebraica (lógica ecuacional multitipada) para modelar estructuras de datos y comportamiento funcional de programas. Los casos donde deben especificarse sistemas reactivos con estados ocultos no observables o los casos donde hay necesidad de modelar tipos de datos infinitos son los que inspiran el uso de especificaciones coalgebraicas (lógica coalgebraica) tomando como fundamento la teoría de coálgebras.

Las coálgebras han sido fundamentales en el desarrollo de múltiples trabajos dentro del área de computación teórica, como el capturar la esencia de la dinámica de sistemas, representar autómatas, sistemas de transición, estructuras de datos infinitas [11], formulación de lenguajes de programación orientada a objetos y especificación formal, en donde esta última usa lógicas modales (coalgebraicas) en el proceso de especificación coalgebraica, constituyendo la esencia del presente documento.

La lógica modal tiene relación con las coálgebras. Esta se puede considerar la lógica natural de la teoría de coálgebras [8], así como la lógica ecuacional lo es para el álgebra universal [5]. En particular, nos interesa la lógica de comportamiento abstracto, en donde las coálgebras de estudio serán las coálgebras provenientes de funtores polinomiales.

La teoría de coálgebras goza de gran importancia en el campo matemático actual, pues permite representar y generalizar objetos establecidos en la teoría de modelos clásica. Una F-álgebra es un par $\langle A, \alpha_A : F(A) \to A\rangle$, donde F es un endofuntor de **Set**. Así, el conjunto de los números naturales con el cero es la F-álgebra, $\langle \mathbb{N}, \alpha_{\mathbb{N}} : F(\mathbb{N}) \equiv 1+\mathbb{N} \to \mathbb{N}\rangle$, donde la definición de $\alpha_{\mathbb{N}}$ corresponde a la elección del cero como $\alpha_{\mathbb{N}}(*) = 0$ y la función sucesor como $\alpha_{\mathbb{N}}(n) = n+1$. De esta forma, la definición de $\alpha_{\mathbb{N}}$ corresponde la escogencia del neutro y la función sucesor. Además, esta F-álgebra es el objeto inicial de la categoría de F-álgebras, lo cual garantiza los principios de inducción y recursión sobre los naturales. Dualmente, la coálgebra $\langle \overline{\mathbb{N}}, \beta_{\overline{\mathbb{N}}} : F(\overline{\mathbb{N}}) \equiv \overline{\mathbb{N}} \to 1+\overline{\mathbb{N}}\rangle$ corresponde a los números naturales extendidos, en donde

$$\beta_{\overline{\mathbb{N}}}(n) = \begin{cases} * & si \quad n = 0, \\ n-1 & si \quad n > 0, \\ \infty & si \quad n = \infty, \end{cases}$$

siendo $\beta_{\overline{\mathbb{N}}}$ la función predecesor. Esta F-coálgebra es el objeto final en la categoría de las F-coálgebras, que proporcionan los principios de correcursión y coinducción.

La descripción de $\mathbb{N}$ y $\overline{\mathbb{N}}$ como F-álgebras y F-coálgebra, respectivamente, no solo tiene importancia en el campo matemático. En ciencias de la computación el poder representar estos objetos usando categorías tiene sus ventajas, pues permite estudiar lenguajes de programación con tipos de datos inductivos empleando las álgebras iniciales, y con tipos de datos coinductivos, empleando las coálgebras finales. Además, con las álgebras iniciales y empleando inducción se pueden construir objetos usando operaciones generadoras, y mediante recursión dan lugar a conjuntos inductivos. Dualmente, la coinducción permite incorporar estructuras infinitas y obtener conjuntos coinductivos, los cuales van a ser representados por coálgebras, en donde sus elementos pueden ser deconstruidos en otros pertenecientes al conjunto empleando operaciones destructoras, a fin de observar su comportamiento estructural viendo sus partes finitas o infinitas.

Análogamente a la lógica algebraica, el estudio de una lógica coalgebraica comprende una componente sintáctica, que corresponde a la interface sintáctica de una especificación y otra semántica, asociada a la interpretación de fórmulas dentro de las coálgebras. Estas componentes deben estar conectadas mediante la relación de satisfacción, que dada su relevancia constitutiva es el mayor reto a establecer. A fin de establecer lógicas en computación que sean independientes del lenguaje, Goguen y Burstall en [7] introducen las instituciones, objetos que abstraen el componente semántico de un sistema lógico relativo a la especificación y en donde las diversas implementaciones dan la posibilidad de cambios de signatura. Es aquí donde un objeto netamente individual como el de signatura se puede ver como un objeto mas complejo, como lo es la categoría de signaturas.

De esta manera, mediante funtores que llamaremos **Mod** y **Sen**, se asociará a cada signatura la clase de álgebras que son modelo de ella y el conjunto de axiomas que se pueden construir a partir de sus símbolos. Con estos tres objetos y la relación de satisfacción entre axiomas y modelos, se definen las instituciones que representan la lógica algebraica y la coinstitución, que representa la lógica coalgebraica siendo en esencia estos dos objetos iguales salvo la definición de los funtores **Mod** y **Sen**, pues mientras en las instituciones el primero es covariante y el segundo contravariante, en las coinstituciones son contravariantes y covariante, respectivamente.

Los componentes y formulación de la lógica ecuacional multitipada [7] y [5] ilustran el camino para modelar una lógica como institución. Otras presentaciones de lógicas como instituciones se pueden consultar en [6].

En nuestro interés particular, nos concentramos en la lógica de comportamiento abstracto [9], que modela sistemas en los que sus estados son equivalentes en comportamiento, es decir, estados que no puede distinguirse a partir de observaciones hechas por un observador externo al sistema, siendo esta relación mas débil que la bisimulación [11]. Aquí presentamos la reconstrucción de esta lógica como una coinstitución coalgebraica. Nos restringiremos a las coálgebras polinomiales para establecer como resultado principal la relación de satisfacción.

En la sección 2 definimos los elementos básicos que conforma una institución, así como la definición del funtor polinomial de interés. En las secciones 3 y 4 abordaremos los elementos básicos que conforman la coinstitución de la lógica de comportamiento abstracto: definimos la categoría de signaturas, la construcción de fórmulas, modelos y finalmente estableceremos la relación de satisfacción como objetivo fundamental en el desarrollo del presente artículo. Finalmente, agradecemos ampliamente al referí anónimo por sus sugerencias, que han mejorado sustancialmente la versión original.

2. Especificación algebraica

Cuando se especifican estructuras de datos o el comportamiento funcional de un sistema de *software*, se emplea la teoría de modelos clásica [2]. Su componente sintáctico, conformado por las ecuaciones, y semántico, conformado por las álgebras, se conectan mediante la relación de satisfacción de la institución algebraica ecuacional, objeto introducido por Goguen y Burstall en [3] y [7] para definir formalmente un sistema lógico. La manera en que están definidas estas instituciones garantizan la invarianza del valor de verdad de toda fórmula bajo cambios de vocabulario. Definiremos las instituciones en un marco general y luego se presentará la definición de su dual, las coinstituciones, objeto que nos permitirá presentar la lógica de comportamiento abstracto.

Definición 2.1. Una institución $\mathcal{I}$ consta de:

I1 Una categoría **Sig** de signaturas;

I2 Un funtor $Sen : \mathbf{Sig} \to \mathbf{Set}$, enviando cada signatura Σ al conjunto $Sen(\Sigma)$ de elementos llamados sentencias sobre esta signatura;

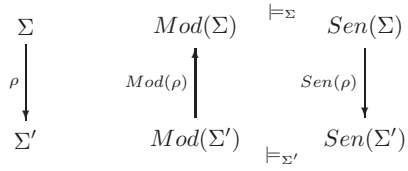
I3 Un funtor $Mod : \mathbf{Sig} \to \mathbf{Cat}^{op}$, que asocia a cada signatura Σ la categoría $Mod(\Sigma)$ de objetos llamados Σ-modelos; y

I4 Una relación $\models_\Sigma \subseteq Mod(\Sigma) \times Sen(\Sigma)$ para cada $\Sigma \in \mathbf{Sig}$, que satisface la siguiente condición de Satisfacción:

Para cada morfismo $\rho : \Sigma \to \Sigma' \in \mathbf{Sig}$, $M' \in Mod(\Sigma')$ y $e \in Sen(\Sigma)$,

$$M' \models_\Sigma Sen(\rho)(e) \quad \Leftrightarrow \quad Mod(\rho)(M') \models_\Sigma e$$

El siguiente diagrama ilustra la condición de satisfacción:



En [3, 7, 5] se presenta la lógica ecuacional multitipada como institución, donde:

- **Sig** tiene como objetos signaturas multitipadas $\langle T, \Sigma \rangle$, donde T es el conjunto de tipos y Σ es una familia de símbolos de operación a los cuales se le asocia una aridad; y morfismos entre signaturas $\varphi : \langle S, \Sigma \rangle \to \langle T, \Sigma' \rangle$, pares de funciones $\varphi_S : S \to T$ y $\varphi_\Sigma : \Sigma \to \Sigma'$, preservando la aridad de cada uno de los símbolos de operación;
- $Sen : \mathbf{Sig} \to \mathbf{Set}$, envía cada signatura Σ al conjunto $Sen(\Sigma) = EQ(\Sigma)$ de ecuaciones;
- $Mod : \mathbf{Sig} \to \mathbf{Cat}^{op}$, envía cada signatura Σ a $\mathbf{Mod}(\Sigma) = \mathbf{Alg}(\Sigma)$, la categoría de las Σ-álgebras y homomorfismos; y
- La relación $\models_\Sigma \subseteq Mod(\Sigma) \times Sen(\Sigma)$ para cada $\Sigma \in \mathbf{Sig}$, es la usual entre ecuaciones y álgebras establecida en el álgebra universal.

Una reconstrucción precisa de esta institución algebraica ecuacional se puede ver en [4]. En [7] se presentan otras lógicas como instituciones. Cuando la signatura consta de un tipo, coincide con la presentada en álgebra universal. Este tipo de signaturas son la base en la definición de los funtores polinomiales, con los que no sólo se obtiene la estructura de las álgebras, sino de las coálgebras. Estos funtores polinomiales tienen la particularidad de que pueden ser construidos con los funtores constante, identidad, suma, producto y potencia [11]. El trabajar con los funtores polinomiales por el lado de las álgebras permite tener álgebras iniciales que garantizan los principios de inducción y recursión. Dualmente, por el lado de coálgebras se tienen coálgebras finales que garantizan los principios de coinducción y correcursión. Además, las coálgebras finales incorporan todos los posibles comportamientos de sistemas determinísticos. Por este motivo hay un gran esfuerzo en el área de computación teórica para mostrar la existencia de coálgebras finales para varios funtores, en donde los polinomiales son uno de ellos. Como el interés son los funtores polinomiales definidos a partir de una signatura, denotamos con $\mathbf{Sig}_E$ la categoría de signaturas de un único tipo. Si $\Sigma \in \mathbf{Sig}_E$, el funtor polinomial $P_\Sigma : \mathbf{Set} \to \mathbf{Set}$ se define como:

1. $P_\Sigma(X) = \coprod_{\sigma\in\Sigma, ar(\sigma)=n} X^n \simeq \bigcup_{\sigma\in\Sigma, ar(\sigma)=n} \{\sigma\} \times X^n$,

   donde $\coprod$ es el operador de unión disjunta . Aquí, los elementos de $P_\Sigma(X)$ son pares $\langle \sigma, (x_i)_{i\leq ar(\sigma)}\rangle$, es decir, $P_\Sigma(X) = \{\langle \sigma, (x_i)_{i\leq ar(\sigma)}\rangle | \sigma \in \Sigma\}$.

2. $f : X \to Y$ es enviada a la función $P_\Sigma(f) : P_\Sigma(X) \to P_\Sigma(Y)$, definida por $P_\Sigma(f)(\langle \sigma, (x_i)_{i\leq ar(\sigma)}\rangle) = \langle \sigma, (f(x_i))_{i\leq ar(\sigma)}\rangle$.

Ejemplo 2.2. Si $\Sigma = \{c^0, \alpha^2\}$, entonces $P_\Sigma(X) = 1 + X \times X$ es un conjunto conformado por el único elemento de 1 ó pares ordenados $(\alpha, (x_1, x_2))$ tales que $x_1, x_2 \in X$ y α es el único símbolo de operación en Σ.

3. Especificación coalgebraica

Los métodos coalgebraicos son apropiados en la especificación de procesos y comportamiento reactivo de un sistema. Las coinstituciones como duales a las instituciones, son convenientes en la representación de lógicas coalgebraicas, en particular, la lógica de comportamiento abstracto que modela propiedades de sistemas determinísticos simples a partir de sus fórmulas, las cuales son subconjuntos de la coálgebra final y que son interpretadas mediante el morfismo final.

Definición 3.1. Una *coinstitución* consta de:

(co1) Una categoría de signaturas Ω;

(co2) Un funtor $Mod : \Omega \to \mathbf{Cat}$;

(co3) Un funtor $Sen : \Omega \to \mathbf{Cat}^{op}$;

(co4) Una familia de relaciones $\models_\Sigma \subseteq Mod(\Sigma) \times Sen(\Sigma)$ indexada por objetos $\Sigma \in \Omega$ que cumple el dual de la condición de satisfacción algebraica

$$Mod(\sigma)(M) \models_\Sigma \phi \quad \Leftrightarrow \quad M \models_\Sigma Sen(\sigma)(\phi)$$

para todo $\sigma : \Sigma \to \Sigma'$, $M \in Mod(\Sigma)$ y $\phi \in Sen(\Sigma')$:

$$\begin{array}{ccc} \Sigma & & Mod(\Sigma) \quad \overset{\models_\Sigma}{} \quad Sen(\Sigma) \\ {\scriptstyle\sigma}\downarrow & & {\scriptstyle Mod(\sigma)}\downarrow \qquad\qquad \uparrow{\scriptstyle Sen(\sigma)} \\ \Sigma' & & Mod(\Sigma') \quad \underset{\models_{\Sigma'}}{} \quad Sen(\Sigma') \end{array}$$

Se destaca la diferencia entre la definición de institución, en donde su funtor Mod es contravariante y Sen covariante, mientras que en las coinstituciones el primero es covariante y el segundo es contravariante, forma conveniente para describir lógicas coalgebraicas. Comenzamos detallando los componentes que conforman la coinstitución coalgebraica que nos permitirá alcanzar el logro de este trabajo, la representación de la lógica de comportamiento abstracto.

3.1. Categoría de signaturas

Si Σ es una signatura de un tipo, es decir, $\Sigma \in \mathbf{Sig}_E$, consideramos en lo que sigue el funtor $P_\Sigma : \mathbf{Set} \to \mathbf{Set}$ como la signatura para las coálgebras. Así, nuestras signaturas no serán conjuntos de símbolos, sino endofuntores polinomiales de **Set**, y los morfismos naturales entre estos objetos corresponderán a las transformaciones naturales. Esta categoría de signaturas coalgebraica se denotará **SIG** y tendrá como objetos endofuntores de **Set** y transformaciones naturales. Uno de los principales motivos por el cual se toma este tipo de funtores es que admiten coálgebras finales dentro de la categoría de coálgebras y nos garantizan la interpretación de fórmulas empleando el morfismo final [10].

3.2. La categoría de P_Σ-coálgebras

Las P_Σ-álgebras son dualizadas de manera natural por las P_Σ-coálgebras, en el sentido de teoría de categorías que surge al cambiar el sentido de la flecha en la estructura de la álgebra. Las coálgebras son importantes en aplicaciones a ciencias de la computación, pues son convenientes para modelar objetos, sistemas de datos, sistemas de transición, sistemas determinísticos, autómatas, especificación de datos, entre otros [11].

Definición 3.2. Si $F : \mathbf{Set} \to \mathbf{Set}$ es un endofuntor sobre **Set**, una *F-coálgebra* es un par (S, α_S), formado por un objeto $S \in \mathbf{Set}$, llamado conjunto de estados, y un morfismo $\alpha_S : S \to F(S)$, llamado función de transición.

Observación 3.3. Todo conjunto X lleva implícita una estructura de coálgebra, considerando P_Σ como el funtor constante $P_\Sigma : \mathbf{Set} \to \mathbf{Set}$, tal que $\Sigma = \{i^0\}$ y $P_\Sigma(X) = 1$, para todo $X \in \mathbf{Set}$, donde $1 = \{*\}$ y $\alpha_X(x) = *$, para todo $x \in X$, siendo $P_\Sigma(X) = X^0 = 1$.

Como se indicó anteriormente, mientras las álgebras construyen elementos a partir de operaciones generadoras, la función de transición en las coálgebras actúa como operación destructora, pues toma un elemento y lo descompone en las partes que lo constituyen siendo en este sentido una función de observaciones, lo que caracteriza las coálgebras y que tiene su principal uso en modelamiento de estructuras de datos infinitas y semántica para lenguajes de programación orientado a objetos [5]. Para notar la diferencia entre álgebras y coálgebras, veamos dos ejemplos típicos dentro del área de computación.

Ejemplo 3.4. Los tipos de datos pueden ser finitos o infinitos. Los tipos de datos finitos se pueden modelar por álgebras. La colección de palabras finitas A^* sobre el alfabeto A, es la P_Σ-álgebra $\langle A^*, \alpha : (1 + (A^* \times A^*)) \to A^* \rangle$, donde α envía el único elemento del conjunto 1 a la palabra vacía, y el par $\langle v, w \rangle$ a su concatenación $v \cdot w$. Esta función es equivalente al siguiente par de funciones: $c : A^* \times A^* \to A^* \quad y \quad d : 1 \to A^*$.

Esto indica que el álgebra de palabras es la terna $\langle A^*, c, d \rangle$ donde $\Sigma = \{c^2, d^0\}$ es la signatura para esta álgebra, coincidiendo con la definición presentada en álgebra universal. Para tipo de datos infinitos las coálgebras son la mejor opción.

Ejemplo 3.5. El conjunto de palabras finitas o infinitas A^∞ sobre el alfabeto A se representa como $\langle A^\infty, \delta : A^\infty \to (1 + (A \times A^\infty)) \rangle$, donde δ envía la palabra vacía al elemento del conjunto 1, y una palabra no vacía al par consistente de su cabeza y cola, es decir,

si $\alpha \in A^{\infty}$ es la palabra $[\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_n, \ldots]$ entonces $\delta(\alpha) = \langle \alpha_0, (\alpha_1, \alpha_2, \ldots \alpha_n, \ldots) \rangle$. Nótese que la coálgebra observa el comportamiento (desarmar) de las listas que están en su dominio mediante la función de transición δ.

Un sistema determinístico es un mecanismo que estando sobre uno de sus estados y recibiendo (no necesariamente) una señal de entrada, cambia su estado a otro y emite (no necesariamente) una señal de salida. Esto indica que cada estado y cada señal de entrada determina un próximo estado y señal de salida. Los sistemas determinísticos [11] más simples de estudiar son los que corresponden a coálgebras polinomiales, los cuales constituyen los funtores de interés en el presente artículo.

Ejemplo 3.6. Si $\Sigma = \{e\}$ con e un símbolo de aridad uno, el funtor polinomial P_Σ es tal que $P_\Sigma(S) = S^{ar(e)} = S$. La coálgebra para este funtor es un sistema determinístico simple cuya función de transición $\alpha_S : S \to S$ indica la dinámica del sistema, definida como $\alpha_S(s) = s'$ y simbolizada $s \to_S s'$, la cual representa la transición que el estado s hace al estado s'. En particular, si queremos especificar el conjunto de los números enteros positivos pares como un sistema determinístico, lo podemos hacer como la coálgebra $(\mathbb{N}, sigpar)$, donde $sigpar$ corresponde a la función *siguiente par* que estaría definida como $sigpar(n) = n + 2$:

$$2 \to 4 \to 6 \to \ldots$$

Las aplicaciones que preservan la estructura de coálgebras corresponden a los homomorfismos entre coálgebras[1].

Definición 3.7. Sean (S, α) y (T, β) dos P_Σ-coálgebras. La aplicación $f : S \to T$ es un homomorfismo entre P_Σ-coálgebras si $P_\Sigma(f) \circ \alpha = \beta \circ f$:

$$\begin{array}{ccc} S & \xrightarrow{f} & T \\ {\scriptstyle\alpha}\downarrow & & \downarrow{\scriptstyle\beta} \\ P_\Sigma(S) & \xrightarrow[P_\Sigma(f)]{} & P_\Sigma(T) \end{array}$$

Ejemplo 3.8. Si $\langle A, \alpha_A \rangle$ y $\langle B, \alpha_B \rangle$ son dos coálgebras que representan los conjuntos A y B respectivamente, como en 3.3, un homomorfismo $f : A \to B$ entre estas dos coálgebras se evidencia por la conmutatividad del siguiente diagrama:

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle\alpha_A}\downarrow & & \downarrow{\scriptstyle\alpha_B} \\ 1 & \xrightarrow[P_\Sigma(f)=i]{} & 1 \end{array}$$

Denotamos como $\mathbf{CoAlg}(P_\Sigma)$ la categoría de P_Σ-coálgebras y P_Σ-homomorfismos. Si P_Σ y $P_{\Sigma'}$ son endofuntores de **Set**, una transformación natural $\sigma : P_\Sigma \to P_{\Sigma'}$ nos permite

[1] Otra forma de relacionar dos coálgebras es mediante la bisimulación [11].

representar una P_Σ-coálgebra como una $P_{\Sigma'}$-coálgebra, mediante la composición de la coestructura de la coálgebra y la componente de σ:

$$A \xrightarrow{\alpha_A} P_\Sigma A \xrightarrow{\sigma(A)} P_{\Sigma'} A$$

El funtor $Mod : \mathbf{SIG} \to \mathbf{Cat}$ de la coinstitución coalgebraica asocia a cada signatura P_Σ la categoría $\mathbf{Coalg}(P_\Sigma)$, y cada morfismo $\sigma : P_\Sigma \to P_{\Sigma'}$ entre signaturas con el funtor covariante $Mod\sigma = \widehat{\sigma} : \mathbf{Coalg}(P_\Sigma) \to \mathbf{Coalg}(P_{\Sigma'})$, que se define de la siguiente forma:

- $\widehat{\sigma}(A, \alpha) = (A, \sigma(A) \circ \alpha)$, donde $(A, \alpha) \in \mathbf{Coalg}(P_\Sigma)$;
- $f : (A, \alpha) \to (B, \beta)$ es enviado a $\widehat{\sigma}(f) : (A, \sigma(A) \circ \alpha_A) \to (B, \alpha_B \circ \sigma(B))$, ilustrado por la conmutatividad del siguiente diagrama:

$$\begin{array}{ccccc} A & \xrightarrow{\alpha_A} & P_\Sigma A & \xrightarrow{\sigma(A)} & P_{\Sigma'} A \\ \downarrow{\scriptstyle f} & & \downarrow{\scriptstyle P_\Sigma f} & & \downarrow{\scriptstyle P_{\Sigma'} f} \\ B & \xrightarrow{\alpha_B} & P_\Sigma B & \xrightarrow{\sigma(B)} & P_{\Sigma'} B \end{array}$$

La presentación del funtor *Sen* requiere la definición de las fórmulas de la lógica de comportamiento abstracto. A continuación indicamos en qué consiste la lógica de comportamiento abstracto, definiendo sus fórmulas y forma de interpretarse.

3.3. Lógica coalgebraica

El estudio de lógicas es inherente a la solución de problemas en ciencias de la computación. Lógicas como la clásica, difusa, de primer orden, ecuacional, entre otras [7], son constituidas por una componente sintáctica y otra semántica relacionadas por una relación de satisfacción. Estas lógicas que pueden variar entre un problema y otro, van ligadas al proceso de especificación, que consiste en determinar las funciones, comportamiento y propiedades presentes en el problema o sistema de *software* a implementar. El especificar estructuras de datos infinitas o procesos internos a una máquina implica usar técnicas coalgebraicas y lógicas que se ajusten a la estructura que modela el problema, siendo estas estructuras en general las coálgebras.

Una lógica para coálgebras de tipo P_Σ es una dupla formada por un lenguaje $\mathcal{L}$ y una familia de relaciones $\models$ indexada por P_Σ-coálgebras. Es decir,

$$\models := \{\models_{(A,\alpha)} : (A, \alpha) \in \mathbf{Coalg}(P_\Sigma)\}, \quad \text{donde } \models_{(A,\alpha)} \subseteq A \times \mathcal{L}, \forall (A, \alpha) \in \mathbf{Coalg}(P_\Sigma).$$

Nos interesa estudiar la lógica de comportamiento, la cual estudia pares de estados que coincidan en algún instante entre las diversas transiciones que puede hacer un sistema. Si (A, α) y (B, β) son objetos en $\mathbf{Coalg}(P_\Sigma)$, dos estados (a, b) en $A \times B$ son *equivalentes en comportamiento*, si existe (E, ϵ) en $\mathbf{Coalg}(P_\Sigma)$ y un par de morfismos $f : (A, \alpha) \to (E, \epsilon)$ y $g : (B, \beta) \to (E, \epsilon)$ tales que $f(a) = g(b)$.

Observación 3.9. Consideremos (A, α) y (T, τ), una coálgebra y la coálgebra final en $\mathbf{Coalg}(P_\Sigma)$, respectivamente, y a, t elementos en A y T. Si consideramos $! : A \to T$ el morfismo final e $i : T \to T$ la identidad sobre T, entonces para que a, t sean equivalentes en comportamiento se debe cumplir que $!(a) = i(t)$. Pero $!(a) = t = i(t) = i(!(a))$. Así, todo estado en la coálgebra (A, α) es equivalente en comportamiento a un estado t en (T, τ).

Se denomina una **lógica de comportamiento** [9], aquella en donde

$$a \models_{(A,\alpha)} \phi \quad \text{si y solo si} \quad b \models_{(B,\beta)} \phi,$$

para todos los estados equivalentes en comportamiento (a, b) y fórmula $\phi \in \mathcal{L}$. Es decir, que los estados equivalentes en comportamiento deben satisfacer la misma fórmula. Por otro lado, $(A, \alpha) \models \phi$ indica que $a \models \phi$ para todo $a \in A$. También, $[\![\phi]\!]_{(A,\alpha)} = \{a \in A : a \models_{(A,\alpha)} \phi\}$ es el conjunto interpretación de una fórmula.

4. Fórmulas

En coálgebra universal las fórmulas se definen a partir de coálgebras finales, y con estas se pueden describir y obtener propiedades de coálgebras para un endofuntor de **Set**. En el campo de ciencias de la computación, las fórmulas son empleadas para describir y observar el comportamiento de los sistemas. Las fórmulas en la lógica de comportamiento se pueden representar semánticamente como subconjuntos de coálgebras finales, las cuales darán cuenta del comportamiento de los estados de ciertos sistemas.

Proposición 4.1. *Suponga que (T, τ) es el objeto final en $\boldsymbol{Coalg}(P_\Sigma)$ y $\mathcal{L}$ una lógica de comportamiento para P_Σ-coálgebras. Entonces,*

$$[\![\phi]\!]_{(A,\alpha)} = !^{-1}([\![\phi]\!]_{(T,\tau)})$$

para toda (A, α) en $\boldsymbol{Coalg}(P_\Sigma)$ y toda $\phi \in \mathcal{L}$, siendo $! : A \to T$ el morfismo final.

Si (T, τ) es una P_Σ-coálgebra final, entonces podemos ver $\mathcal{P}(T)$ como una lógica de comportamiento, con la siguiente regla de interpretación: $a \models_{(A,\alpha)} \phi$ si $!(a) \in \phi$, donde $\phi \in \mathcal{P}(T)$ y $!$ es el morfismo final $! : A \to T$.

Definición 4.2. Supóngase que (T, τ) es un objeto final en $\mathbf{Coalg}(P_\Sigma)$. La lógica de *comportamiento abstracto* $\mathcal{A}(P_\Sigma) = \mathcal{P}(T)$ tiene como fórmulas subconjuntos de la P_Σ-coálgebra final, y la satisfacción de fórmulas por coálgebras viene dada por la condición: $a \models_{(A,\alpha)} \phi$ si $!(a) \in \phi$.

Finalmente, el funtor $Sen : \mathbf{SIG} \to \mathbf{Set}^{op}$ envía la signatura P_Σ al conjunto de fórmulas $\mathcal{A}(P_\Sigma)$, y un morfismo de signaturas $\sigma : P_\Sigma \to P_{\Sigma'}$ a la función $Sen\sigma : \mathcal{A}(P_{\Sigma'}) \to \mathcal{A}(P_\Sigma)$, definida como:

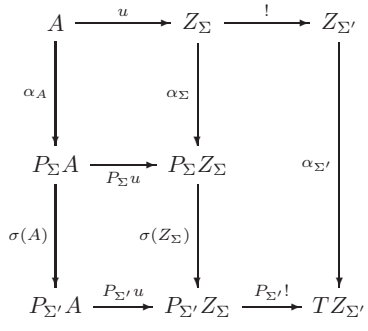
- $\mathcal{A}(P_{\Sigma'}) \supseteq X \rightarrowtail Sen\sigma(X) = !^{-1}(X)$, donde $! : Mod\sigma(T_\Sigma, \tau_\Sigma) \to (T_{\Sigma'}, \tau_{\Sigma'})$ es el morfismo final en $\mathbf{Coalg}(P_{\Sigma'})$.

Teniendo los funtores Mod y Sen, podemos presentar la condición de satisfacción [9] de la coinstitución de la lógica de comportamiento abstracto.

Lema 4.3. *Sea $\sigma : P_\Sigma \to P_{\Sigma'}$ una transformación natural, donde P_Σ y $P_{\Sigma'}$ admiten coálgebras finales $\langle Z_\Sigma, \alpha_\Sigma\rangle$ y $\langle Z_{\Sigma'}, \alpha_{\Sigma'}\rangle$, respectivamente. Entonces*

$$\langle A, \alpha_A\rangle \models !^{-1}(\phi) \;\Leftrightarrow\; \widehat{\sigma}\langle A, \alpha_A\rangle \models \phi$$

para toda $\langle A, \alpha_A\rangle \in \boldsymbol{Coalg}(P_\Sigma)$ y todo $\phi \in \mathcal{A}(P_{\Sigma'})$, para el único morfismo $! : \widehat{\sigma}\langle Z_\Sigma, \alpha_\Sigma\rangle \to \langle Z_{\Sigma'}, \alpha_{\Sigma'}\rangle$ dado por la finalidad:



Teorema 4.4. *Sea **SIG** la categoría de signaturas polinomiales, Mod y Sen los funtores definidos anteriormente y la condición de satisfacción del lema anterior. Entonces la* **lógica de comportamiento abstracto** *LCA es la coinstitución dada por $\mathcal{I}_{LCA} = \langle \boldsymbol{SIG}, Mod, Sen, \models\rangle$.*

Referencias

[1] Awodey S., *Category Theory*, Oxford Logic Guides, 49. The Clarendon Press, Oxford University Press, New York, 2006.

[2] Burris S. and Sankappanavar H., *A Course In Universal Algebra*, Graduate Texts in Mathematics, 78. Springer-Verlag, New York-Berlin, 1981.

[3] Burstall R. and Goguen J., "The Semantics of Clear, A Specification Languaje", in *Proc. Copenagen Winter School on Abstract Software Specifications*, (ed. Dines Bjorner), Springer, Vol. 86 (1979), 292-332.

[4] Castaño Perea J.A., "Especificación coalgebraica ecuacional y coecuacional", Tesis (M.Sc.), Universidad del Valle, Cali, Colombia, 2011, 36 p.

[5] Domínguez C., "Especificación orientada a objetos de sistemas de cálculo simbólico", Tesis (Ph.D), Universidad de la Rioja, España, Logroño, 2003, 70 p.

[6] Goguen J. and Burstall R., "A study in the foundations of programming methodology: specifications, institutions, charters and parchments", Category theory and computer programming (Guildford, 1985), 313-333, *Lecture Notes in Comput. Sci.*, 240, Springer, Berlin, 1986.

[7] Goguen J. and Burstall R., "Institutions: abstract model theory for specification and programming", *J. Assoc. Comput. Mach.* 39 (1992), no. 1, 95-146.

[8] Kurz A., "A co-variety-theorem for modal logic. Advances in modal logic", Vol. 2 (Uppsala, 1998), 367-380, *CSLI Lecture Notes*, 119, CSLI Publ., Stanford, CA, 2001

[9] Pattinson D., "Translating Logics for Coalgebras", in *Lecture Notes in Computer Science*, Recent Trends in Algebraic Development Techniques, Springer, Vol. 2755 (2003), 393-408.

[10] Pierce B., *Basic Category Theory for Computer Scientist*, Londres, Inglaterra, The MIT Press 1991.

[11] Rutten J.J.M.M., "Universal coalgebra: a theory of systems. Modern algebra and its applications", (Nashville, TN, 1996), *Theoret. Comput. Sci.* 249 (2000), no. 1, 3-80.

[12] MacLane S., *Categories for the working mathematician*, Graduate Texts in Mathematics, Vol. 5. Springer-Verlag, New York-Berlin, 1971.