MARGARITA MATHEMATICA EN MEMORIA DE
JOSÉ JAVIER (CHICHO) GUADALUPE HERNÁNDEZ
(Luis Español y Juan L. Varona, editores),
Servicio de Publicaciones, Universidad de La Rioja,
Logroño, Spain, 2001.

INSTITUCIONES: MATEMÁTICAS PARA LA ESPECIFICACIÓN EN COMPUTACIÓN

CÉSAR DOMÍNGUEZ, LAUREANO LAMBÁN, VICO PASCUAL Y JULIO RUBIO

Dedicado a Chicho

Abstract. In this paper we study formally an operation on signatures. This operation assigns to each signature Σ a signature Σ_{imp} in such a way that a Σ_{imp}-model corresponds to a family of Σ-models. This construction has allowed us to analyse the data structures which were used in a symbolic computation system called EAT (a software created by F. Sergeraert to calculate the homology of iterated loop spaces). To this end we introduce the notions of institution and institution morphism and we present the components of the equational algebraic institution. Finally we show that the operation $(\ )_{imp}$ induces (can be extended to) an endomorphism of the equational algebraic institution.

1. Introducción

Hace unos años, los autores del presente trabajo nos empeñamos en el estudio formal de las estructuras de datos que aparecen en un programa denominado EAT (Effective Algebraic Topology). Este es un software diseñado por Sergeraert [14] y dedicado al Cálculo Simbólico en Topología Algebraica. En particular, está especializado para el cálculo de homología de espacios de lazos iterados. Desde el punto de vista de la práctica de la computación, la característica más destacable de este programa es la necesidad de construir y manipular, en tiempo de ejecución, estructuras de datos de naturaleza potencialmente infinita (objetos localmente efectivos según la terminología usada en [13]). Esto constituye una diferencia sustancial respecto de otros sistemas de Cálculo Simbólico en otras áreas de las Matemáticas distintas de la Topología Algebraica o el Álgebra Homológica. El modo de implementación usado en EAT para estas estructuras de datos fue estudiado en [9], obteniendo caracterizaciones de las implementaciones usadas en EAT en términos de objetos finales en adecuadas categorías de implementaciones.

En este trabajo apenas se incide en cuestiones relacionadas con la implementación. Por contra, el estudio que se realiza se centra fundamentalmente en aspectos que pertenecen al ámbito de la especificación. El trabajo se ha dividido en tres secciones. En la sección 2 se introduce la noción de institución. Una institución es un marco en el que realizar especificaciones. Así, la definición de institución abstrae

2000 *Mathematics Subject Classification.* 68Q65.

Key words and phrases. Institution, algebraic specification, symbolic computation.

Parcialmente subvencionado por DGES, proyecto PB98-1621-C02-01, y por Universidad de La Rioja, proyecto API-00/B28.

las componentes básicas necesarias en cualquier marco de especificación. Para ilustrar esta definición se presenta una institución particular: la institución algebraica ecuacional. En la sección 3 se introduce una operación general entre signaturas que viene a formalizar el patrón que siguen las estructuras de datos en EAT. A partir de cada signatura Σ se construye una signatura Σ_{imp} que, desde una perspectiva computacional, puede entenderse como la signatura que define a las familias de implementaciones de modelos de Σ. En el trabajo se muestra la existencia de objeto final en adecuadas subcategorías de Σ_{imp}-álgebras. Además, estos objetos finales son los modelos implementados en EAT. Por último, en la sección 4 se extiende esta operación entre signaturas a un morfismo de instituciones, en particular a un endomorfismo de la institución algebraica ecuacional.

Este trabajo pretende servir para que el lector obtenga una cierta perspectiva de los temas en los que estamos investigando. Por ello, hemos procurado evitar los resultados y detalles técnicos y, por contra, hemos premeditadamente abundado en la explicación de las definiciones y en la presentación de ejemplos. Serán los lectores conocidos los que nos indiquen hasta qué punto hemos conseguido nuestro propósito.

2. Instituciones

En la producción de software en general y en el desarrollo de un tipo de datos en particular, deben separarse dos partes que no siempre resultan bien diferenciadas en la práctica: especificación e implementación. En la especificación tiene que quedar bien definido en qué consiste ese tipo de datos, es decir, cuál es su dominio y cuáles son sus operadores. Por su parte, la implementación debe proporcionar una realización del tipo: representación adecuada para el dominio y métodos que lleven a cabo las operaciones. Así, la especificación recoge toda la información que los posibles usuarios necesitan: les permitirá declarar datos del tipo y manipularlos con los operadores sin posibilidad ni necesidad de acceder a los detalles propios de la implementación.

Desde la década de los setenta se vienen usando técnicas algebraicas para la especificación de tipos de datos. El lenguaje del álgebra permite definir sin ambigüedad el conjunto de valores que constituyen el dominio de un tipo de datos y el comportamiento de sus operadores. La teoría de especificación algebraica se ha desarrollado notablemente y se han estudiado diferentes técnicas de especificación (por ejemplo, en [12], [5]).

A continuación vamos a presentar las componentes básicas de cualquier marco de especificación formal. Para ello, comenzaremos por describir brevemente uno de ellos, el conocido como *especificación algebraica ecuacional*.

Una *signatura* Σ es un par de conjuntos de símbolos $\Sigma = (S, \Omega)$, los elementos de S se denominan *géneros* y los elementos de Ω *operaciones*. Cada $\omega \in \Omega$ tiene asociada una *aridad*, es decir, una secuencia no vacía de elementos de S, $s_1 \dots s_n s$, que viene a representar el perfil de la operación. En este caso, ω se refiere a una operación con n argumentos (cada uno de ellos del género correspondiente) y resultado de género s. Un *morfismo* $h : \Sigma \to \Sigma'$ entre signaturas $\Sigma = (S, \Omega)$ y $\Sigma' = (S', \Omega')$ es una pareja de aplicaciones conjuntistas $h = (h_1 : S \to S', h_2 : \Omega \to \Omega')$ compatible con los perfiles de las operaciones, es decir, si $\omega \in \Omega$ tiene aridad $s_1 \dots s_n s$, su imagen, $h_2(\omega)$, tiene aridad $h_1(s_1)\dots h_1(s_n)h_1(s)$. Signaturas y morfismos de signaturas definen una categoría SIG que constituye el ámbito sintáctico de la especificación algebraica ecuacional.

Ilustraremos con unos ejemplos estas definiciones. Consideramos una signatura `MON` con un único género $S = \{m\}$ y cuyas operaciones son $\Omega = \{e : m,\ bin : mmm\}$. Usando un lenguaje apropiado para describir signaturas tenemos:

signatura `MON`
  generos m
  operaciones $e :\to m$
    $bin : m\ m \to m$
finsig

Ejemplos característicos de morfismos de signaturas son los renombrados de géneros y operaciones. Por ejemplo, si tomamos la signatura:

signatura `MON2`
  generos l
  operaciones $e :\to l$
    $concat : l\ l \to l$
finsig

la pareja $\Phi \equiv (\Phi_1, \Phi_2)$ dada por $\Phi_1(m) = l,\ \Phi_2(e) = e,\ \Phi_2(bin) = concat$ define un morfismo de signaturas que, obviamente, resulta ser un isomorfismo. Consideramos ahora la signatura definida por:

signatura `GRP`
  generos g
  operaciones $e :\to g$
    $prd : g\ g \to g$
    $inv : g \to g$
finsig

Entre `MON` y `GRP` podemos considerar el morfismo $\eta \equiv (\eta_1, \eta_2)$ dado por: $\eta_1(m) = g,\ \eta_2(e) = e,\ \eta_2(bin) = prd$. El morfismo η corresponde a otro tipo de morfismo natural entre signaturas: la inclusión de signaturas.

Dada una signatura $\Sigma = (S, \Omega)$, una *ecuación* en Σ es una terna $e \equiv (X, t_1, t_2)$ donde X es un conjunto de variables (cada $x \in X$ tiene asociado un género $s \in S$) y t_1, t_2 son términos (fórmulas sintácticamente correctas) sobre Σ con variables en X. Por simplicidad, no se suele indicar explícitamente el conjunto de variables X, por lo que una ecuación se suele expresar de la forma $t_1 = t_2$. Serían ecuaciones en `MON`:

$bin(x, e) = bin(e, x)$
$bin(x, e) = x$
$bin(x, bin(y, z)) = bin(bin(x, y), z)$

Es claro que un morfismo de signaturas $\Phi : \Sigma \to \Sigma'$ transforma Σ-términos en Σ'-términos. Esta construcción permite definir un funtor *sentencias*, $Sen : SIG \to SET$, donde SET denota la categoría de conjuntos, que asocia a cada signatura Σ el conjunto de ecuaciones en Σ.

En el ámbito de la especificación algebraica ecuacional, una *especificación* consiste en una pareja $ESPEC = (\Sigma, E)$, siendo Σ una signatura y E un conjunto de ecuaciones en Σ. El siguiente paso será establecer un criterio para determinar la veracidad de las ecuaciones. Para ello, será necesario considerar los «sitios» (modelos) en los que poder interpretar las ecuaciones.

Fijada una signatura $\Sigma = (S, \Omega)$, una Σ-*álgebra* $\mathbb{A}$ es un par de familias $\mathbb{A} = ((A_s)_{s\in S}, (\omega^{\mathbb{A}})_{\omega\in\Omega})$. Para cada $s \in S$, A_s es un conjunto, llamado *soporte* en $\mathbb{A}$ del género s, y para cada $\omega \in \Omega$ de perfil $\omega : s_1 \dots s_n s$, $\omega^{\mathbb{A}}$ es una función $\omega^{\mathbb{A}} : A_{s_1} \times \cdots \times A_{s_n} \to A_s$, llamada *interpretación* en $\mathbb{A}$ de la operación ω. Así, una Σ-álgebra no es sino un posible modelo matemático para la sintaxis definida por Σ. Por su parte, un *morfismo* $f : \mathbb{A} \to \mathbb{B}$ de Σ-álgebras es una familia de aplicaciones $(f_s : A_s \to B_s)_{s\in S}$ que hacen conmutativos todos los diagramas asociados a las operaciones $\omega \in \Omega$. Las Σ-álgebras y los Σ-morfismos definen una categoría, que denotaremos $Alg(\Sigma)$.

Si nos fijamos en la signatura `GRP`, una `GRP`-álgebra consiste en un conjunto que tiene un elemento distinguido (correspondiente a la operación $e :\to g$) y que dispone de dos operaciones internas, una unaria y otra binaria. En concreto, cualquier grupo resulta ser una `GRP`-álgebra. Además, los homomorfismos de grupos son morfismos de `GRP`-álgebras. Análogamente, cualquier monoide es una `MON`-álgebra.

Dado un morfismo de signaturas $\Phi : \Sigma \to \Sigma'$ y una Σ'-álgebra $\mathbb{A}$, consideramos la Σ-álgebra $\mathbb{B}$ que tiene como soporte de cada género s, $B_s = A_{\Phi_1(s)}$, y como interpretación de $\omega \in \Omega$ la operación en $\mathbb{A}$ que interpreta a $\Phi_2(\omega)$, es decir, $\omega^{\mathbb{B}} = \Phi_2(\omega)^{\mathbb{A}}$. Esta construcción se extiende a un funtor $Mod : SIG \to CAT^{OP}$, donde CAT^{OP} denota la dual de la categoría de categorías. Por ejemplo, la inclusión de la signatura `MON` en `GRP` induce el funtor olvido entre la categoría de `GRP`-álgebras y la de `MON`-álgebras.

La última de las componentes básicas de un marco de especificación va a ser una relación entre sentencias y modelos. En nuestro caso particular, la relación que determina si un modelo satisface o no una sentencia. Dadas una Σ-álgebra $\mathbb{A}$ y una Σ-ecuación $e \equiv t_1 = t_2$, se dice que $\mathbb{A}$ *satisface* e, lo denotamos $\mathbb{A} \models e$, si para cada valoración en $\mathbb{A}$ de las variables de e se cumple la igualdad $t_1 =_{\mathbb{A}} t_2$ (fijados los valores de las variables, cada Σ-término admite una única interpretación en una Σ-álgebra $\mathbb{A}$). De modo natural, la relación de satisfacibilidad se extiende a conjuntos de ecuaciones: $\mathbb{A} \models E$ si $\mathbb{A}$ satisface todas las ecuaciones del conjunto E. Así, una especificación $ESPEC = (\Sigma, E)$ determina una subcategoría de $Alg(\Sigma)$, la subcategoría plena generada por aquellas Σ-álgebras que satisfacen E.

Siguiendo con el ejemplo, la signatura `GRP` se extiende a una especificación:

ESPEC `GRP`
generos g
operaciones $e :\to g$
$prd : g\ g \to g$
$inv : g \to g$
ecuaciones $prd(x, e) = x$
$prd(e, x) = x$
$prd(prd(x, y), z) = prd(x, prd(y, z))$
$prd(x, inv(x)) = e$
$prd(inv(x), x) = e$
finespec

La categoría de modelos de esta especificación es la categoría de grupos. De modo análogo, como subcategoría de $Alg(\mathtt{MON})$ se obtendría la categoría de monoides.

Éstas son las cuatro componentes que constituyen un marco abstracto de especificación, lo que habitualmente aparece en la literatura con el nombre de institución: signaturas, sentencias, modelos y relación de satisfacibilidad.

Definición 2.1. Una *institución* $\mathcal{I}$ viene dada por:

1. Una categoría SIG, llamada *categoría de signaturas* de $\mathcal{I}$.
2. Un funtor $Sen : SIG \to SET$, llamado *funtor sentencias*.
3. Un funtor $Mod : SIG \to CAT^{OP}$, llamado *funtor modelos*.
4. Una clase de relaciones $\models \subseteq (Mod(\Sigma) \times Sen(\Sigma))_{\Sigma \in Objetos(SIG)}$ de forma que, para cada morfismo $h : \Sigma \to \Sigma'$ en SIG, $\mathbb{A}' \in Mod(\Sigma')$ y $e \in Sen(\Sigma)$ se tiene:
$$Mod(h)(\mathbb{A}') \models e \quad \text{si y sólo si} \quad \mathbb{A}' \models Sen(h)(e).$$

El ejemplo que hemos desarrollado para ilustrar la noción de institución, que se denomina habitualmente *institución algebraica ecuacional*, constituye el formalismo más extendido para la especificación de tipos de datos. De hecho, una buena parte de las instituciones en las que habitualmente se trabaja son generalizaciones o variantes del marco algebraico ecuacional. Por ejemplo, para poder especificar tipos de datos en los que algunos de los operadores sean parciales, podemos incorporar a las especificaciones de la institución algebraica ecuacional sentencias de definitud (y considerar algunas ecuaciones como condicionales), lo que modificando adecuadamente la relación de satisfacibilidad, dará lugar a una institución en la que los modelos serán álgebras parciales. Otras posibles generalizaciones se obtienen al considerar diferentes tipos de sentencias (lógica de predicados, por ejemplo).

Un marco que resulta de gran interés por su adecuación para formalizar el modo de trabajar en el paradigma de la programación orientada a objetos es la institución de las especificaciones ocultas. Eliminando detalles técnicos, la idea fundamental consiste en distinguir en las signaturas dos tipos de géneros (géneros visibles y géneros ocultos) y fijar un modelo para la parte visible de la especificación. Esto significa en la práctica que en los modelos vamos a poder distinguir lo que son los soportes de datos de lo que son los soportes para los objetos.

Referencias adecuadas sobre instituciones son [6], [1]. Para especificaciones ocultas [7].

3. La construcción $(\ )_{imp}$

En algunos sistemas de software para el Cálculo Simbólico, en concreto en EAT (Effective Algebraic Topology) [14] y su sucesor Kenzo [4] que fueron diseñados para el cálculo de grupos de homología y homotopía de espacios topológicos, se hace necesaria la construcción y manipulación (en tiempo de ejecución) de estructuras de datos de naturaleza infinita. Por ejemplo, construcción de conjuntos simpliciales o de complejos de cadenas como objetos intermedios necesarios durante la ejecución de algunos de los algoritmos de cálculo. El modo en el que estas estructuras fueron implementadas en EAT fue analizado en [9] obteniendo como resultados más destacados:

1. Estas estructuras admiten especificaciones que pueden ser interpretadas como especificaciones de implementaciones de tipos de datos más básicos. Por tanto, sus implementaciones corresponden a lo que podríamos denominar «implementaciones al cuadrado».
2. Las implementaciones dadas en EAT para estas estructuras son, en cierto sentido, canónicas. Resultan ser objetos finales en adecuadas categorías de implementaciones.

En esta sección se incide fundamentalmente sobre aspectos relacionados con las especificaciones de las estructuras de datos de EAT. El lector interesado en el modo de implementación y en el estudio teórico realizado sobre el mismo puede consultar [9], [10], [3].

Aunque no representa adecuadamente la complejidad de algunos de los tipos de datos que aparecen en EAT y Kenzo, el ejemplo de grupo nos permite ilustrar el tipo de especificaciones con las que se trabaja en dichos programas. Un grupo tiene asociada la signatura `GRP` presentada en la sección anterior:

signatura `GRP`
  generos g
  operaciones $e :\to g$
    $prd : g\ g \to g$
    $inv : g \to g$
finsig

Ahora bien, para poder construir un grupo en tiempo de ejecución deberemos considerar un tipo de datos cuyas instancias sean grupos, cada dato nos debe permitir acceder a la información asociada a un grupo, es decir, como mínimo recuperar las operaciones de un grupo. Para ello podemos considerar la signatura:

signatura $\mathtt{GRP}_{imp}$
  generos grp, $elem$
  operaciones $imp\_e : grp \to elem$
    $imp\_prd : grp\ elem\ elem \to elem$
    $imp\_inv : grp\ elem \to elem$
finsig

De este modo, en cada modelo $\mathbb{A}$ de $\mathtt{GRP}_{imp}$, cada elemento z de A_{grp} (soporte del género grp), nos permite recuperar las operaciones de una `GRP`-álgebra mediante la terna

$$(imp\_e^{\mathbb{A}}(z), imp\_prd^{\mathbb{A}}(z,-,-), imp\_inv^{\mathbb{A}}(z,-)).$$

La idea que subyace es que el género *grp* representa a los grupos, más concretamente a implementaciones de grupos, mientras que el género *elem* representa a los elementos de los grupos. Este patrón de especificación se aproxima adecuadamente a las estructuras de datos usadas en EAT y, como veremos a continuación, responde a un tipo de construcción con buenas propiedades. La relación entre las signaturas `GRP` y $\mathtt{GRP}_{imp}$ es un caso particular de la operación entre signaturas que a continuación se describe.

Sea SIG el conjunto de signaturas sobre un alfabeto concreto. Definimos $(\ )_{imp} : SIG \to SIG$ la aplicación que, a cada signatura $\Sigma = (S, \Omega)$ le asocia la signatura $\Sigma_{imp} = (S_{imp}, \Omega_{imp})$ dada por

- $S_{imp} = \{imp_\Sigma\} \cup S$, donde imp_Σ es un símbolo nuevo que no está en S.
- $\Omega_{imp} = \{imp\_\omega : imp_\Sigma s_1 \dots s_n s\}_{(\omega: s_1 \dots s_n s \in \Omega)}$. Por cada operación ω de aridad $s_1 \dots s_n s$ se incluye en Ω_{imp} una operación imp_ω cuya aridad es $imp_\Sigma s_1 \dots s_n s$.

En [9] puede encontrarse una extensión de esta construcción como operación entre tipos abstractos de datos.

Los siguientes ejemplos muestran la relación existente entre Σ-álgebras y Σ_{imp}-álgebras.

Ejemplo 3.1. Definimos la $\mathtt{GRP}_{imp}$-álgebra $\mathbb{A}$ dada por: $\mathbb{Z}$ es el soporte en $\mathbb{A}$ para el género *elem* y $\mathbb{N}$ el soporte del género *grp*. La interpretación en $\mathbb{A}$ de las operaciones es:

$imp\_unt^{\mathbb{A}}(n) = 0$
$imp\_prd^{\mathbb{A}}(n, z1, z2) = z1 + z2$
$imp\_inv^{\mathbb{A}}(n, z) = -z$

De esta forma, cada $n \in \mathbb{N}$ determina una `GRP`-álgebra que denotamos A_n, siendo $A_n = \langle \mathbb{Z}, \{+_{\mathbb{Z}}, -_{\mathbb{Z}}, 0_{\mathbb{Z}}\}\rangle$. La $\mathtt{GRP}_{imp}$-álgebra $\mathbb{A}$ admite ser descrita como una colección indexada por $\mathbb{N}$ de copias de $\mathbb{Z}$.

Ejemplo 3.2. Consideramos $\mathbb{B}$ con los mismos soportes del ejemplo anterior y operaciones:

$imp\_unt^{\mathbb{B}}(n) = 0$
$imp\_prd^{\mathbb{B}}(n, z1, z2) = (z1 + z2)\ mod\ n$
$imp\_inv^{\mathbb{B}}(n, z) = -z\ mod\ n$

Para cada $n \in \mathbb{N}$, obtenemos la `GRP`-álgebra $B_n = \langle \mathbb{Z}, \{+_{mod\ n}, -_{mod\ n}, 0_{\mathbb{Z}}\}\rangle$ que tiene el mismo comportamiento que el grupo $\mathbb{Z}/n\mathbb{Z}$ (notar que B_n no es isomorfo a $\mathbb{Z}/n\mathbb{Z}$, de hecho ni siquiera es grupo, pero existe un cociente del mismo que sí lo es).

Ejemplo 3.3. Consideramos la misma signatura $\mathtt{GRP}_{imp}$ y para cada $n \in \mathbb{N}$, $n > 1$, fijo consideramos el conjunto $\langle n \rangle = \{0, 1, \dots, n-1\}$. Definimos la $\mathtt{GRP}_{imp}$-álgebra $\mathbb{C}$ tomando $\langle n \rangle$ como soporte para *elem* y, para el género *grp*, el conjunto $C_{grp} = \{(n_1, \dots, n_k) : k \geq 1, n_i > 1\ \forall i \in \{1, \dots, k\},\ n = n_1 * \dots * n_k\}$. La idea subyacente en la elección de los soportes de $\mathbb{C}$ es que cada tupla $(n_1, \dots, n_k)$ de C_{grp} representa al grupo abeliano finito $\mathbb{Z}/n_1\mathbb{Z}\times\cdots\times\mathbb{Z}/n_k\mathbb{Z}$. Por tanto, el conjunto C_{grp} está dando soporte a una familia de grupos abelianos finitos.

Para completar la definición de la $\texttt{GRP}_{imp}$-álgebra $\mathbb{C}$ hay que introducir las operaciones. Teniendo en cuenta la última observación, las operaciones de $\mathbb{C}$ se definirán a partir de las de cada grupo $\mathbb{Z}/n_1\mathbb{Z}\times\cdots\times\mathbb{Z}/n_k\mathbb{Z}$ y, por tanto, será suficiente con dar una biyección entre el conjunto $\langle n\rangle$ y los elementos del grupo $\mathbb{Z}/n_1\mathbb{Z}\times\cdots\times\mathbb{Z}/n_k\mathbb{Z}$. Cada elemento del grupo $\mathbb{Z}/n_1\mathbb{Z}\times\cdots\times\mathbb{Z}/n_k\mathbb{Z}$ puede representarse por una tupla $(a_1,\ldots,a_k)$ con $a_i\in\{0,\ldots,n_i-1\}$, para todo $i=1,\ldots,k$. Consideramos la biyección $enum_{(n_1,\ldots,n_k)}\colon \mathbb{Z}/n_1\mathbb{Z}\times\cdots\times\mathbb{Z}/n_k\mathbb{Z}\to\{0,1,\ldots,n-1\}$ definida del siguiente modo:

$$enum_{(n_1,\ldots,n_k)}((a_1,\ldots,a_k))=\sum_{i=1}^{k}\Big(\prod_{j=i+1}^{k}n_j\Big)a_i.$$

Esta biyección nos sirve para definir las operaciones de $\mathbb{C}$, lo que completa la definición de la $\texttt{GRP}_{imp}$-álgebra $\mathbb{C}$.

Por ejemplo, si $n=12$, la tupla $(2,2,3)$ representa al grupo $\mathbb{Z}/2\mathbb{Z}\times\mathbb{Z}/2\mathbb{Z}\times\mathbb{Z}/3\mathbb{Z}$. La biyección $enum_{(n_1,\ldots,n_k)}$ lleva a cada tupla (i,j,k) con $i,j\in\{0,1\}$ y $k\in\{0,1,2\}$ al natural $6*i+3*j+k\in\langle n\rangle$. La tupla $(2,6)$ representa al grupo $\mathbb{Z}/2\mathbb{Z}\times\mathbb{Z}/6\mathbb{Z}$ y $(4,3)$ a $\mathbb{Z}/4\mathbb{Z}\times\mathbb{Z}/3\mathbb{Z}$. Observar que entre los tres grupos anteriores, los dos primeros son isomorfos.

Cualquier grupo abeliano finito es isomorfo a un $\mathbb{Z}/n_1\mathbb{Z}\times\cdots\times\mathbb{Z}/n_k\mathbb{Z}$, con $n_i\in\mathbb{N}$, $\forall i=1,\ldots,k$. Por tanto, fijado $n\in\mathbb{N}$, la familia definida en este ejemplo cubre todos los grupos abelianos finitos de cardinal n. Barriendo todos los $n\in\mathbb{N}$, se cubren todos los grupos abelianos finitos. Luego este ejemplo nos aporta un patrón común de representación para los grupos abelianos finitos.

Además, desde cualquier representación de grupos abelianos definidos sobre $\langle n\rangle$ existe, al menos, un $\texttt{GRP}_{imp}$-morfismo a la $\texttt{GRP}_{imp}$-álgebra $\mathbb{C}$. Un tal $\texttt{GRP}_{imp}$-morfismo puede definirse de manera sencilla siempre que los elementos de $\langle n\rangle$ queden fijos.

Ejemplo 3.4. La $\texttt{GRP}_{imp}$-álgebra $\mathbb{C}$ anterior nos permite representar familias de grupos abelianos finitos. Vamos ahora a dar una $\texttt{GRP}_{imp}$-álgebra, que llamaremos $\mathbb{D}$ y que recoge el caso de los grupos finitos cualesquiera.

Fijamos un $n\in\mathbb{N}$ y, como en el ejemplo anterior, tomamos el conjunto $\langle n\rangle$ como soporte para el género *elem*. Consideramos como soporte para el género *grp* el conjunto $D_{grp}=\{$ matrices $n\times n$: la matriz es la tabla del producto de un grupo sobre $\langle n\rangle\}$. De cada tabla, pueden extraerse los inversos de cada elemento así como el elemento neutro, por lo que la $\texttt{GRP}_{imp}$-álgebra $\mathbb{D}$ representa de forma canónica a una familia de grupos.

Cualquier grupo finito es isomorfo a uno de los grupos indexados por el conjunto D_{grp} anterior, por lo que, barriendo todos los $n\in\mathbb{N}$, se cubren todos los grupos finitos.

Por ejemplo, fijamos $n=6$ y consideramos el grupo de permutaciones de tres elementos que denotamos por $\mathcal{S}_3$. Vamos a suponer, por concretar, que los tres elementos permutados son $1,2$ y 3. Si asignamos a cada permutación un ordinal según el orden lexicográfico, es decir, si tomamos la biyección $enum_{\mathcal{S}_3}\colon\mathcal{S}_3\to\langle n\rangle$ dada por: $enum_{\mathcal{S}_3}((1\ 2\ 3))=0$, $enum_{\mathcal{S}_3}((1\ 3\ 2))=1$, $enum_{\mathcal{S}_3}((2\ 1\ 3))=2$,

$enum_{\mathcal{S}_3}((2\ 3\ 1)) = 3$, $enum_{\mathcal{S}_3}((3\ 1\ 2)) = 4$ y $enum_{\mathcal{S}_3}((3\ 2\ 1)) = 5$, la siguiente matriz representa la tabla de multiplicar del grupo:

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 0 & 3 & 2 & 5 & 4 \\ 2 & 4 & 0 & 5 & 1 & 3 \\ 3 & 5 & 1 & 4 & 0 & 2 \\ 4 & 2 & 5 & 0 & 3 & 1 \\ 5 & 3 & 4 & 1 & 2 & 0 \end{pmatrix}.$$

Desde cualquier posible representación de grupos definidos sobre $\langle n \rangle$ existe un único $\mathtt{GRP}_{imp}$-morfismo a la $\mathtt{GRP}_{imp}$-álgebra $\mathbb{D}$, manteniendo fijo el soporte $\langle n \rangle$. Por ello, la $\mathtt{GRP}_{imp}$-álgebra $\mathbb{D}$ es la más «general» entre las representaciones de grupos sobre $\langle n \rangle$. Esto se traducirá en una propiedad de finalidad del álgebra $\mathbb{D}$ en una categoría adecuada.

En particular, la $\mathtt{GRP}_{imp}$-álgebra $\mathbb{D}$ da una representación para grupos abelianos finitos y, para cada $n \in \mathbb{N}$ fijo, el $\mathtt{GRP}_{imp}$-morfismo de $\mathbb{C}$ a $\mathbb{D}$ que mantiene fijo $\langle n \rangle$ viene dado por una aplicación que a cada tupla $(n_1, \dots, n_k) \in C_{grp}$ le asocia una matriz $n \times n$. Por ejemplo, si consideramos $n = 6$, $C_{grp} = \{(2,3), (3,2), (6)\}$. Para asociar una matriz a cada uno de los elementos de C_{grp} utilizamos la biyección $enum_{(n_1,\dots,n_k)}$, y así obtenemos que

$$f_{grp}((2,3)) = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 0 & 4 & 5 & 3 \\ 2 & 0 & 1 & 5 & 3 & 4 \\ 3 & 4 & 5 & 0 & 1 & 2 \\ 4 & 5 & 3 & 1 & 2 & 0 \\ 5 & 3 & 4 & 2 & 0 & 1 \end{pmatrix}, \quad f_{grp}((3,2)) = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 0 & 3 & 2 & 5 & 4 \\ 2 & 3 & 4 & 5 & 0 & 1 \\ 3 & 2 & 5 & 4 & 1 & 0 \\ 4 & 5 & 0 & 1 & 2 & 3 \\ 5 & 4 & 1 & 0 & 3 & 2 \end{pmatrix}.$$

Como es fácil observar, los ejemplos anteriores corresponden todos ellos a la siguiente construcción:

Proposición 3.5. *Dada una signatura $\Sigma = (S, \Omega)$ y una Σ_{imp}-álgebra*

$$\mathbb{A} = \langle A_{imp_\Sigma},\ (A_s)_{s\in S},\ \{imp\_\omega^{\mathbb{A}} : A_{imp_\Sigma} \times A_{s_1} \times \cdots \times A_{s_n} \to A_s\}_{\omega\in\Sigma}\rangle,$$

cada elemento $a \in A_{imp_\Sigma}$ define una Σ-álgebra B_a de la siguiente forma:

$$B_a = \langle (A_s)_{s\in S},\ \{imp\_\omega^{\mathbb{A}}(a, \_) : A_{s_1} \times \cdots \times A_{s_n} \to A_s\}_{\omega\in\Sigma}\rangle.$$

Conviene notar que todas las Σ-álgebras B_a de la familia anterior tienen los mismos soportes, es decir, una Σ_{imp}-álgebra se corresponde con una familia indexada de Σ-álgebras en la que todas las álgebras de la familia tienen los mismos soportes. Este hecho, que a nivel de modelos no es demasiado relevante, es de gran importancia a nivel de las implementaciones. En el ejemplo de la estructura grupo, no es razonable implementar todos los grupos, sino los grupos cuyos elementos tienen un mismo patrón, un mismo modo de representación en el computador.

Llevando las ideas del comentario anterior al nivel de los modelos, dada una signatura $\Sigma = (S, \Omega)$ no conviene tratar de modo conjunto toda la categoría de modelos $Alg(\Sigma_{imp})$. Así, fijado un dominio $D = (D_s)_{s\in S}$, consideramos la subcategoría de

$Alg(\Sigma_{imp})$ cuyos objetos son las Σ_{imp}-álgebras que tienen a la familia D como soportes de los géneros de S (dejando libre el soporte del género añadido imp_{Σ}) y cuyos morfismos son aquellos Σ_{imp}-morfismos que son la identidad sobre los géneros de S. A esta subcategoría la denotaremos $Alg^D(\Sigma_{imp})$. Es conveniente notar que, para cada dominio $D = (D_s)_{s \in S}$, la correspondiente categoría $Alg^D(\Sigma_{imp})$ no resulta ser una subtegoría plena de $Alg(\Sigma_{imp})$. El lector interesado puede encontrar un estudio más detallado de estas categorías $Alg^D(\Sigma_{imp})$ y su relación con categorías de familias indexadas sobre $Alg(\Sigma)$ en [10] y [11].

Entre todas las álgebras de $Alg^D(\Sigma_{imp})$ hay una que resulta especialmente relevante. Consideramos $\mathbb{A}^{can}$ dada por:

- En los géneros $s \in S$, $A_s^{can} = D_s$.
- En el género imp_{Σ}, $A_{imp_{\Sigma}}^{can}$ es el conjunto de tuplas funcionales $(f_1, \ldots, f_k)$, una función por cada operación de Ω, de forma que $\langle (D_s)_{s \in S}, (f_1, \ldots, f_k) \rangle$ es una Σ-álgebra.
- La interpretación de $\omega_i \in \Omega_{imp}$ con perfil $\omega_i : imp_{\Sigma} s_1 \ldots s_n \to s$ es

$$A_{\omega_i}^{can}((f_1, \ldots, f_k), d_1, \ldots, d_n) = f_i(d_1, \ldots, d_n).$$

Como demuestra el siguiente resultado, el álgebra $\mathbb{A}^{can}$ es un modelo con buenas propiedades y, por tanto, un candidato adecuado para ser implementado como modelo de $Alg^D(\Sigma_{imp})$.

Teorema 3.6. *La Σ_{imp}-álgebra $\mathbb{A}^{can}$ es objeto final de la categoría $Alg^D(\Sigma_{imp})$.*

4. La construcción $(\ )_{imp}$ como morfismo de instituciones

Como ya ha sido comentado, la noción de institución pretende establecer las componentes necesarias para cualquier marco en el que realizar especificaciones. De modo natural aparece la cuestión de establecer pasos entre diferentes instituciones, de forma que se puedan trasladar especificaciones de una institución a otra. Esto conduce al concepto de morfismo de instituciones [6].

Definición 4.1. Dadas dos instituciones $\mathcal{I}$ y $\mathcal{I}'$, un *morfismo de instituciones* $\Phi : \mathcal{I} \to \mathcal{I}'$ consiste en:

1. Un funtor $\Phi : SIG \to SIG'$ entre las correspondientes categorías de signaturas.
2. Una transformación natural $\alpha : Sen' \circ \Phi \Rightarrow Sen$, es decir, para cada signatura Σ de SIG una aplicación $\alpha_{\Sigma} : Sen'(\Phi(\Sigma)) \to Sen(\Sigma)$, transformando sentencias de $\Phi(\Sigma)$ en sentencias de Σ de forma compatible con los morfismos de signaturas.
3. Una transformación natural $\beta : Mod \Rightarrow Mod' \circ \Phi$. Esta transformación vendrá dada por una clase de funtores $\beta_{\Sigma} : Mod(\Sigma) \to Mod'(\Phi(\Sigma))$, uno para cada signatura Σ.
4. Las transformaciones α y β respetan las relaciones de satisfacibilidad de $\mathcal{I}$ y de $\mathcal{I}'$. Esto es, se cumple
$$A \models \alpha_{\Sigma}(e') \quad \text{si y sólo si} \quad \beta_{\Sigma}(A) \models' e'$$
para cualquier Σ-modelo A en $\mathcal{I}$ y cualquier $\Phi(\Sigma)$-sentencia e' en $\mathcal{I}'$.

De modo informal, un morfismo de instituciones proporciona relaciones entre las diferentes componentes de una institución: signaturas, sentencias y modelos, respetando las relaciones de satisfacibilidad.

En lo que sigue, se va a probar que la construcción $(\ )_{imp}$ presentada en la sección anterior se puede extender a un morfismo de instituciones. En concreto, lo que aquí se desarrolla es una presentación de $(\ )_{imp}$ como endomorfismo de la institución algebraica ecuacional. Otras posibilidades son estudiadas en [2] considerando esta construcción como un morfismo de instituciones entre la institución algebraica ecuacional y la institución de las especificaciones ocultas.

Dada la institución algebraica ecuacional $\mathcal{I} = (SIG, Sen, Mod, \models)$ tal y como ha sido descrita en la segunda sección de este trabajo, definimos el endomorfismo de $\mathcal{I}$ dado por:

- Un funtor $\Phi : SIG \to SIG$ que asocia a cada signatura Σ su correspondiente signatura Σ_{imp}. Recordar que Σ_{imp} tiene un género añadido a los géneros de Σ, que denotamos imp_Σ, y que este nuevo género es agregado como primer argumento a todas las operaciones $\omega : s_1 \dots s_n \to s$ de Σ dando lugar a las operaciones de Σ_{imp}. Dado un morfismo de signaturas $h : \Sigma \to \Sigma'$ en SIG, $\Phi(h) : \Sigma_{imp} \to \Sigma'_{imp}$ es el morfismo cuya restricción a los géneros de Σ es h y tal que $\Phi(h)(imp_\Sigma) := imp_{\Sigma'}$. Respecto de las operaciones, $\Phi(h)(imp\_\omega) := imp\_\omega'$, con $\omega' = h(\omega)$.
- Una transformación natural $\alpha : Sen \circ \Phi \Rightarrow Sen$ definida por la familia de aplicaciones $\alpha_\Sigma : Sen(\Sigma_{imp}) \to Sen(\Sigma)$, una para cada objeto Σ de SIG, tal que la imagen por α_Σ de una sentencia de Σ_{imp} se obtiene eliminando todos los términos del género imp_Σ (que necesariamente deberán ser variables) y sustituyendo cada operación imp_ω de Σ_{imp} por la correspondiente operación ω de Σ.

  Para que la aplicación anterior esté bien definida es necesario excluir del conjunto de sentencias aquéllas que son identidades entre variables (en concreto, entre variables del género imp_Σ). No obstante, no se pierde generalidad al hacer esta restricción. Las ecuaciones de la forma $z = z$ no aportan significación sobre los modelos. Una ecuación del tipo $z = z'$, de variables de un género s, se puede eliminar construyendo una especificación equivalente en la que los términos de género s sean identificados con una única constante.
- Una transformación natural $\beta : Mod \Rightarrow Mod \circ \Phi$ definida por la familia de funtores $\beta_\Sigma : Mod(\Sigma) \to Mod(\Sigma_{imp})$, uno para cada signatura Σ. Sobre los objetos de $Mod(\Sigma)$, la imagen por β_Σ de una Σ-álgebra $\mathbb{A}$ es la Σ_{imp}-álgebra que tiene al conjunto unipuntual $\{*\}$ como soporte del género imp_Σ y el resto de los soportes los mismos de $\mathbb{A}$. La interpretación de las operaciones de Σ_{imp} en $\beta_\Sigma(\mathbb{A})$ es la natural:

  $$imp\_\omega^{\beta_\Sigma(\mathbb{A})}(*, d_1, \dots, d_n) = \omega^{\mathbb{A}}(d_1, \dots, d_n).$$

  Respecto de los morfismos de $Mod(\Sigma)$, dado un morfismo de Σ-álgebras $f = (f_s)_{s\in S} : \mathbb{A} \to \mathbb{B}$, su imagen $\beta_\Sigma(f)$ es $(id_{\{*\}}, (f_s)_{s\in S})$.

Resulta fácil comprobar que (Φ, α, β) define un endomorfismo de la institución algebraica ecuacional, obteniéndose el siguiente resultado.

Teorema 4.2. *Dada la institución algebraica ecuacional* $\mathcal{I} = (SIG, Sen, Mod, \models)$, *modificada prohibiendo las ecuaciones que identifican variables, la terna* (Φ, α, β) *definida anteriormente constituye un endomorfismo de* $\mathcal{I}$.

Para terminar el artículo, queremos indicar que las matemáticas que aquí aparecen (expresadas al uso habitual, en forma de proposiciones y teoremas) no están tan alejadas de la práctica cotidiana de la programación como podría parecer. De hecho, uno de los objetivos de nuestra línea de investigación consiste en analizar sistemas de cálculo simbólico *reales*, como EAT o Kenzo, que están en funcionamiento en varias universidades. Una de las formas de vincular estas matemáticas a los programas reales es a través de la noción de *tipo abstracto de datos* (y, más en concreto, mediante el concepto de *implementación* de un tipo abstracto de datos [8], [9]). Un tipo abstracto de datos viene dado por una signatura Σ y una clase $\mathcal{C}$ de Σ-álgebras cerradas por isomorfismo. De este modo, una especificación $ESPEC = (\Sigma, E)$ define un tipo abstracto de datos, la clase de Σ-álgebras que satisfacen las ecuaciones de E.

En el endomorfismo anterior queda implícita una operación entre especificaciones. Dada una especificación $ESPEC = (\Sigma, E)$ de $\mathcal{I}$, se puede construir otra especificación $ESPEC_{imp} = (\Sigma_{imp}, E_{imp})$ donde cada ecuación e' de E_{imp} se obtiene a partir de una ecuación e de E, cuantificando con una variable z del nuevo género imp_{Σ} y aplicando la siguiente recursión en los subtérminos de e:

- Dejar sin modificación los subtérminos que sean variables.
- Sustituir los subtérminos $\omega(t_1, \ldots, t_n)$ por $imp\_\omega(z, t'_1, \ldots, t'_n)$, donde cada t'_i es el transformado, por recurrencia, de t_i.

Esta nueva especificación verifica la propiedad esperada, que generaliza la proposición 3.5: si $\mathbb{A}$ es una $ESPEC_{imp}$-álgebra, cada elemento del soporte $A_{imp_{\Sigma}}$ define una $ESPEC$-álgebra.

Referencias

[1] R. Burstall y R. Diaconescu, Hiding and behaviour: An institutional approach, en *A classical mind: Essays in honour of C. A. R. Hoare*, Prentice-Hall (1994), 75–92.

[2] C. Domínguez, L. Lambán, M. V. Pascual y J. Rubio, Hidden specification of a functional system, en *Proceedings Eurocast'2001*, Lecture Notes in Computer Science **2178** (2001).

[3] C. Domínguez y J. Rubio, Modelling inheritance as coercion in a symbolic computation system, en *Proceedings ISSAC'2001*, ACM Press (2001), 109–115.

[4] X. Dousson, F. Sergeraert y Y. Siret, *The Kenzo program*, `http://www-fourier.ujf-grenoble.fr/~sergerar/Kenzo/`, Institut Fourier, Grenoble, 1999.

[5] H. Ehrig y B. Mahr, *Fundamentals of algebraic specification 1. Equations and initial semantics*, Springer-Verlag, 1985.

[6] J. Goguen y R. Burstall, Institutions: abstract model theory for specification and programming, *J. Assoc. Comput. Mach.* **39** (1992), 95–146.

[7] J. Goguen y G. Malcolm, A hidden agenda, *Theoret. Comput. Sci.* **245** (2000), 55–101.

[8] C. Hoare, Proofs of correctness of data representations, *Acta Informatica* **1** (1972), 271–281.

[9] L. Lambán, V. Pascual y J. Rubio, Specifying implementations, en *Proceedings ISSAC'99*, ACM Press (1999), 245–251.

[10] L. Lambán, V. Pascual y J. Rubio, Simplicial sets in the EAT system, en *Proceedings EACA'99*, Universidad de La Laguna (1999), 267–276.
[11] L. Lambán, V. Pascual y J. Rubio, An object-oriented interpretation of the EAT system, prepublicación.
[12] J. Loeckx, H. D. Ehrich y M. Wolf, *Specification of abstract data types*, Wiley-Teubner, 1996.
[13] J. Rubio y F. Sergeraert, Locally effective objects and algebraic topology, en *Computational Algebraic Geometry*, Birkhäuser (1993), 235–251.
[14] J. Rubio, F. Sergeraert y Y. Siret, *EAT: symbolic software for effective homology computation*, `ftp://fourier.ujf-grenoble.fr/pub/EAT`, Institut Fourier, Grenoble, 1997.

Departamento de Matemáticas y Computación, Universidad de La Rioja, Edificio Vives, Calle Luis de Ulloa s/n, 26004 Logroño, Spain
Correo electrónico: {`cedomin,lalamban,mvico,jurubio`}`@dmc.unirioja.es`