Capítulo 4

Especificação de Tipos Abstractos de Dados

Um *tipo de dados* é uma colecção de domínios de elementos, designados elementos básicos, e de operações nesses domínios, de tal forma que todos os elementos dos domínios podem ser gerados a partir dos elementos básicos através das operações. Os domínios de elementos são enumeráveis.

Um *tipo abstracto de dados* é uma classe de tipos que é fechada para a re-designação dos domínios de elementos, dos elementos, e das operações, consequentemente é independente da representação.

Vamos ver que é possível, e é apropriado, escrever uma especificação algébrica equacional para especificar os tipos abstractos de dados. Vai-se construir a álgebra de termos quociente T_{Esp}, que é *gerada* pelos símbolos de operação, e que satisfaz uma dada equação fechada *eq* se e só se *eq* é válida em todas as Esp-álgebras. Esta propriedade designa-se por *típica.* As propriedades de ser *gerada*, e de ser *típica* caracterizam, de forma única a menos de um isomorfismo, T_{Esp}.

4.1 Um exemplo: O TAD Cadeia de Caracteres

O tipo abstracto de dado *Cadeia de Caracteres* tem como elementos as sequências finitas de elementos de um dado alfabeto de base.

Em termos matemáticos os elementos deste TAD são palavras $x_1, \ldots, x_m$ de comprimento $n \geq 0$. Para $n = 0$ tem-se a palavra vazia, para $n > 0$ é uma sequência de elementos de um dado alfabeto A.

É necessário definir antes de mais o alfabeto $A = \{a_1, \ldots, a_n\}$. As palavras são elementos do domínio A^*, o conjunto de todas as sequências enumeráveis de elementos de A, e que inclue a sequência vazia. Como operações temos a *constrói*, que para todo o elemento de A constrói uma sequência de comprimento 1, *concatena*, que dadas duas palavras constrói um outra palavra por justaposição das duas palavras dadas como argumento, *aesq* que adiciona um novo elemento à esquerda de uma dada palavra, e *adir* que adiciona um novo elemento à direita de uma dada palavra.

Temos então o seguinte modelo matemático:

$$\text{SeqCar} = (\{A, A^*\}, \{a_1, a_2, \ldots, a_n, \text{vazia}, \text{constroi}, \text{concatena}, \text{aesq}, \text{adir}\})$$

com

$$\begin{array}{rcccll}
a_i & : & 1 & \longrightarrow & A & \\
 & & * & \longmapsto & a_i, & \text{para todo o } 0 \leq i \leq n \\
\text{vazia} & : & 1 & \longrightarrow & A^* & \\
 & & * & \longmapsto & \text{vazia} & \\
\text{constroi} & : & A & \longrightarrow & A^* & \\
 & & a & \longmapsto & \text{constroi}(a) = a & \\
\text{concatena} & : & A^* \times A^* & \longrightarrow & A^* & \\
 & & (u, v) & \longmapsto & uv & \\
\text{aesq} & : & A \times A^* & \longrightarrow & A^* & \\
 & & (a, u) & \longmapsto & au & \\
\text{adir} & : & A^* \times A & \longrightarrow & A^* & \\
 & & (u, a) & \longmapsto & ua &
\end{array}$$

De seguida temos uma especificação algébrica equacional, `SEQCAR`, para qual *SeqCar* é uma `SEQCAR`-álgebra, e na qual o alfabeto é considerado uma sub-espécie das sequências de caracteres.

```
module SEQCAR {
 [Alfabeto < Palavras]
 ops a1,...,an : -> Alfabeto
 op vazia : -> Palavras
 op constroi : Alfabeto -> Palavras
 op concatena : Palavras Palavras -> Palavras {assoc id: vazia}
 op aesq : Alfabeto Palavras -> Palavras
 op adir : Palavras Alfabeto -> Palavras
 var A : Alfabeto
 var S : Palavras
 eq aesq(A,S) = concatena(constroi(A),S) .
 eq adir(S,A) = concatena(S,constroi(A)) .
}
```

A especificação `SEQCAR` e a álgebra *SeqCar* definem o mesmo Tipo Abstracto de Dados SeqCar, que vamos designar por *TAD(SeqCar)*, e que consiste na álgebra *SeqCar* assim como em todas as álgebras que lhe são isomorfas, em especial a álgebra de termos quociente T_{SeqCar} que pode ser construída a partir da especificação equacional `SEQCAR`.

4.2 Álgebra de Termos Quociente

Trata-se de construir, para cada especificação equacional `ESP`, uma dada `ESP`-álgebra. Dado que a construção se vai efectuar através da definição de classes quociente de termos da assinatura `Esp`, designa-se esta álgebra por *álgebra de termos quociente*, T_{Esp}.

Antes de definir de uma forma geral o que se entende, e como se pode construir, uma álgebra de termos quociente vamos ver o exemplo concreto de `SEQCAR`, isto é T_{SeqCar}.

T_{SeqCar}, a álgebra de termos quociente para a especificação `SEQCAR`.

1º passo: Definição de conjuntos de termos para cada uma das espécies consideradas na especificação.

$$T_{\mathrm{Alfabeto}} = T_{OP,\mathrm{Alfabeto}}; \quad T_{\mathrm{Palavras}} = T_{OP,\mathrm{Palavras}}$$

estes conjuntos são definidos indutivamente

1. $a_1, \ldots, a_n \in T_{\mathrm{Alfabeto}}, \quad vazia \in T_{\mathrm{Palavras}}$
2. Para todo o $ta \in T_{\mathrm{Alfabeto}}$ e $ts_1, ts_2 \in T_{\mathrm{Palavras}}$ então constroi(ta), $concatena(ts_1, ts_2)$, $aesq(ta, ts_1)$, $adir(ts_1, ta) \in T_{\mathrm{Palavras}}$

2º passo: Estabelecer uma relação de equivalência nos termos. Dois termos são ditos equivalentes

$$t_1 \equiv t_2$$

se e só se a valoração $\mathrm{val}_A : T_{OP} \to A$ aplicada a ambos os termos dá o mesmo resultado

$$\mathrm{val}_A(t_1) = \mathrm{val}_A(t_2) \quad \text{para todas as ESP-álgebras } A.$$

A presente definição de relação de equivalência entre termos tem como consequência que todos os termos que coincidam com os lados esquerdos e direitos de equações são equivalentes entre si. Por exemplo:

$$\mathrm{concatena}(\mathrm{vazia}, \mathrm{vazia}) \equiv \mathrm{vazia}$$

pela 1ª equação da especificação equacional.

Além disso para toda a derivação $t_1 \stackrel{*}{\Rightarrow} t_2$ tem-se que $\mathrm{val}(t_1) = \mathrm{val}(t_2)$ e consequentemente $t_1 \equiv t_2$.

Uma classe de equivalência pode ser considerada como um elemento abstracto. Os representantes de cada uma das classes vão ser os elementos da álgebra de termos quociente.

$$[t] = \{t_1 \mid t_1 \equiv t\} \quad \text{para todo o } t \in T_{OP} = T_{\text{Alfabeto}} \cup T_{\text{Palavras}}$$

Os *conjuntos quociente* são então definidos do seguinte modo

$$Q_{\text{Alfabeto}} = \{[t] \mid t \in T_{\text{Alfabeto}}\}$$

$$Q_{\text{Palavras}} = \{[t] \mid t \in T_{\text{Palavras}}\}$$

estes são os conjuntos portadores para a seguinte SEQCAR-álgebra, T_{SeqCar}, que é a álgebra de termos quociente para a especificação SEQCAR.

$$\begin{aligned} T_{\text{SeqCar}} = \big(\{Q_{\text{Alfabeto}}, Q_{\text{Palavras}}\}, \{&a_{1_Q}, \ldots, a_{n_Q}, \text{vazia}_Q, \text{constroi}_Q, \\ &\text{concatena}_Q, \text{aesq}_Q, \text{adir}_Q\}\big) \end{aligned}$$

com

$$a_{i_Q} = [a_i], \; i = 1, \ldots, n$$

$$\text{vazia}_Q = [\text{vazia}]$$

$$\begin{array}{rrcl}
\text{constroi}_Q\colon & Q_{\text{Alfabeto}} & \longrightarrow & Q_{\text{Palavras}} \\
& [a] & \longmapsto & \text{constroi}_Q([a]) = [\text{constroi}(a)] \\
\text{concatena}_Q\colon & Q_{\text{Palavras}} \times Q_{\text{Palavras}} & \longrightarrow & Q_{\text{Palavras}} \\
& ([ts_1], [ts_2]) & \longmapsto & \text{concatena}_Q([ts_1], [ts_2]) = \\
& & & = [\text{concatena}(ts_1, ts_2)] \\
\text{aesq}_Q\colon & Q_{\text{Alfabeto}} \times Q_{\text{Palavras}} & \longrightarrow & Q_{\text{Palavras}} \\
& ([a, ts_1]) & \longmapsto & \text{aesq}_Q([a, ts_1]) = \\
& & & = [\text{aesq}(a, ts_1)] \\
\text{adir}_Q\colon & Q_{\text{Palavras}} \times Q_{\text{Alfabeto}} & \longrightarrow & Q_{\text{Palavras}} \\
& ([ts_1, a]) & \longmapsto & \text{adir}_Q([ts_1, a]) = \\
& & & = [\text{adir}(ts_1, a)]
\end{array}$$

para todo o $a \in T_{\text{Alfabeto}}$ e $ts_1, ts_2 \in T_{\text{Palavras}}$.

3º passo: Para verificar que a álgebra T_{SeqCar} definida deste modo é uma SEQCAR-álgebra é necessário verificar que:

1. as funções estão bem definidas;
2. a álgebra T_{SeqCar} satisfaz as equações em SEQCAR.

As funções estão bem definidas significa, por exemplo para a função constroi, que para $ta \equiv ta'$ então $\text{constroi}(ta) \equiv \text{constroi}(ta')$.

Pela definição da relação de equivalência $\equiv$, isto significa que:

$$\text{val}_A(ta) = \text{val}_A(ta') \Rightarrow \text{val}_A(\text{constroi}(ta)) = \text{val}_A(\text{constroi}(ta'))$$

ora

$$\begin{aligned}
\text{val}_A(\text{constroi}(ta)) &= \text{val}_A(\text{constroi})(val_A(ta)) \\
&= \text{constroi}_A(val_A(ta)) \\
&= \text{constroi}_A(val_A(ta')) \qquad \text{por hipótese} \\
&= \text{val}_A(\text{constroi}(ta'))
\end{aligned}$$

ou seja a relação de equivalência $\equiv$ é uma congruência, isto é, é compatível com as operações.

As álgebras de termos quociente definidas deste modo satisfazem as equações, vejamos que assim é num caso concreto.

Para $[ts] \in Q_{\text{Palavras}}, [ta] \in Q_{\text{Alfabeto}}$, temos:

$$\begin{aligned}
\text{adir}_Q([ts],[ta]) &= [\text{adir}(ts,ta)], \\
&\qquad \text{por definição de adir}_Q \\
&= [\text{concatena}(ts, constroi(ta)], \\
&\qquad \text{dado que para toda a SeqCar-álgebra } A \text{ temos} \\
&\qquad\quad \begin{aligned} \text{val}_A(\text{adir}(ts,ta)) &= \text{adir}_A(\text{val}_A(ts), \text{val}_A(ta)) \\ &= \text{concatena}_A(\text{val}_A(ts), \text{constroi}_A(\text{val}(ta))) \\ &= \text{val}_A(\text{concatena}(ts, \text{constroi}(ta))) \end{aligned} \\
&= \text{concatena}_Q([ts], \text{constroi}_Q([ta])), \\
&\qquad \text{pela definição de concatena}_Q\text{, e de constroi}_Q.
\end{aligned}$$

Este facto advém do facto de que a relação de equivalência, tal como foi definida, é uma congruência nos termos, isto é, respeita (é compatível com) as operações.

Definição 4.1 (Congruência nos termos de base) *Dado uma especificação* Esp $= (E, OP, Eq)$, *a relação de equivalência* $\equiv$ *definida nos termos fechados (de base) por:*

$$t_1 \equiv t_2 \text{ sse } \text{val}_A(t_1) = \text{val}_A(t_2), \qquad t_1, t_2 \in T_{OP} \textit{ e } A \textit{ uma } \text{Esp-}\textit{álgebra.}$$

é designada por congruência nos termos de base

A congruência nos termos de base satisfaz um conjunto de condições, sendo que algumas delas são comuns a todas as relações de equivalência, enquanto outras já são específicas da relação definida acima.

Proposição 4.1 (Propriedades da Congruência) *A congruência nos termos de base satisfaz as seguintes condições, para quaisquer termos em* T_{OP}*:*

1. $t_1 \equiv t_1$, *(reflexividade)*
2. $t_1 \equiv t_2$ *implica* $t_2 \equiv t_1$, *(simetria)*
3. $t_1 \equiv t_2$ *e* $t_2 \equiv t_3$ *implica* $t_1 \equiv t_3$, *(transitividade)*
4. $t_1 \equiv t_1', \ldots, t_n \equiv t_n'$ *implica* $N(t_1, \ldots, t_n) \equiv N(t_1' \ldots, t_n')$, *(congruência)* *para todo o símbolo de operação* $N : e_1 \ldots e_n \to e, n \geq 1$ *e* $N \in OP$
5. *Cada derivação* $t_1 \overset{*}{\Rightarrow} t_2$ *em* Eq *entre termos fechados* t_1, t_2, *implica* $t_1 \equiv t_2$.
6. *Se existe uma* Esp-*álgebra* A *com* $\mathrm{val}(t_1) \neq \mathrm{val}(t_2)$ *para quaisquer* t_1, t_2 *termos fechados em* T_{OP} *então* $t_1 \not\equiv t_2$.

A demonstração destes factos encontra-se em (Ehrig & Mahr, 1985).

Temos agora a definição de Álgebra de Termos Quociente.

Definição 4.2 (Álgebra de Termos Quociente) *Dado uma especificação* $\mathrm{Esp} = (E, OP, Eq)$, *e dada a congruência nos termos de base, a álgebra de termos quociente é definida por:*

1. *para cada espécie* $e \in E$ *tem-se o conjunto portador*

$$Q_e = \{[t] \mid t \in T_{OP,e}\}$$

   com $[t] = \{t' \mid t' \equiv t\}$;

2. *para cada símbolo de constante* $c : \mathbf{1} \to e$ *em* OP *a constante* c_Q *é a classe de congruência gerada por* c

$$c_Q = [c]$$

3. *para cada símbolo de operação* $N : e_1 \ldots e_n \to e$ *em* OP *a operação* $N_Q : Q_{e_1} \times \ldots \times Q_{e_n} \to Q_e$ *é definida por*

$$N_Q([t_1], \ldots, [t_n]) = [N(t_1, \ldots, t_n)]$$

   para todos os termos t_i *de espécie* e_i, *para* $i = 1, \ldots, n$.

Para poder "comparar" álgebras é necessário explicitar como é que se pode definir funções entre álgebras. A noção de homomorfismo, uma função que preserva a estrutura, surge neste contexto como forma de relacionar duas álgebras.

Definição 4.3 (Homomorfismo e Isomorfismo) *Sejam A e B duas álgebras com a mesma assinatura $\Sigma = (E, OP)$.*

Homomorfismo *Um homomorfismo $f : A \to B$, também designado por um Σ-homomorfismo, é uma família de funções $f_e : A_e \to B_e$ para $e \in E$ tal que:*

- *para cada símbolo de constante $c : \mathbf{1} \to e$ em OP, e com $e \in E$ tem-se:*
$$f_e(c_A) = c_B$$
- *para cada símbolo de operação $N : e_1 \ldots e_2 \to e$ em OP, e para todo o $a_i \in A_{e_i}$, com $i = 0, \ldots, n$, tem-se:*
$$f_e(N_A(a_1, \ldots, a_n)) = N_B(f_{e_1}(a_1), \ldots, f_{e_n}(a_n))$$

Isomorfismo *Um homomorfismo $f : A \to B$ é dito um isomorfismo, $f : A \simeq B$ se todas as funções $f_e : A_e \to B_e$, para todo o $e \in E$, são bijectivas.*

Álgebras isomorfas *As álgebra A e B são ditas isomorfas, $A \simeq B$, se existe um isomorfismo $f : A \simeq B$, ou $g : B \simeq A$.*

No seguimento temos um resultado que estabelece algumas propriedades das álgebras de termos quociente, e que permite ver como elas se relacionam com o tipos abstractos de dados.

Teorema 4.1 (Propriedades da Álgebra de Termos Quociente) *A álgebra de termos quociente T_{Esp} de uma especificação $\mathrm{Esp} = (E, OP, Eq)$ tem as seguintes propriedades:*

Gerada: *a valoração $\mathrm{val} : T_{OP} \to T_{\mathrm{Esp}}$ é igual a $\mathrm{nat} : T_{OP} \to T_{\mathrm{Esp}}$, definida por $\mathrm{nat}(t) = [t]$ para todo o $t \in T_{OP}$, como consequência val é sobrejectiva.*

Típica: *cada equação $e = (t_1, t_2)$ de termos fechados $t_1, t_2 \in T_{OP}$ é válida em T_{Esp} sse é válida em todas as Esp-álgebras A.*

T_{Esp} é uma Esp-álgebra.

Antes de demonstrar este resultado é importante referir que a álgebra de termos quociente é (pode-se demonstrar) a única Esp-álgebra que verifica as propriedades de ser *Gerada* e *Típica*. Mais a álgebra de termos quociente pode ser vista como um objecto inicial na categoria de todas as Esp-álgebras.

Demonstração.

I) Demonstra-se que $\mathrm{val} = \mathrm{nat} : T_{OP} \to T_{\mathrm{Esp}}$ por indução estrutural, e usando a definição de val e de T_{Esp}.

Para todo o símbolo de constante $c : \mathbf{1} \to e$ em OP tem-se

$$\mathrm{val}(c) = c_Q = [c] = \mathrm{nat}(c)$$

Para todo o símbolo de operação $N : e_1 \dots e_n \to e$ em OP com $n \geq 1$ e termos $t_i \in T_{OP,e_i}$ com $\mathrm{val}(t_i) = \mathrm{nat}(t_i)$ para $i = 1, \dots, n$ tem-se

$$\begin{aligned}
\mathrm{val}(N(t_1, \dots t_n)) &= N_Q(\mathrm{val}(t_1), \dots, \mathrm{val}(t_n)) \\
&= N_Q(\mathrm{nat}(t_1), \dots, \mathrm{nat}(t_n)) \\
&= N_Q([t_1], \dots, [t_n]) \\
&= [N_Q(t_1, \dots, t_n)] \\
&= \mathrm{nat}(N_Q(t_1, \dots, t_n))
\end{aligned}$$

temos então que $\mathrm{val} = \mathrm{nat}$. O facto de nat ser sobrejectivo vem como consequência de

$$T_{\mathrm{Esp},e} = Q_e = \{[t] \mid t \in T_{OP,e}\}$$

II) Para todo o $t_1, t_2 \in T_{OP,e}$, as seguintes afirmações são equivalentes:

1. $e = (t_1, t_2)$ é válida em T_{Esp}.

   $e = (t_1, t_2)$ é válida em T_{Esp} se para todas as atribuições $\mathrm{atr} : X \to T_{\mathrm{Esp}}$ se tem que $\overline{\mathrm{atr}}(t_1) = \overline{\mathrm{atr}}(t_2)$. Mas para $X = \emptyset$ tem-se que $\overline{\mathrm{atr}} = \mathrm{val}$ e neste caso temos que $\mathrm{val}(t_1) = \mathrm{val}(t_2)$, como já vimos em I.

2. $\mathrm{val}(t_1) = \mathrm{val}(t_2)$

   pela propriedade I, pela definição de nat, e pela definição de $[t]$.

3. $\mathrm{nat}(t_1) = \mathrm{nat}(t_2)$

   pela definição de nat, e pela definição de $[t]$.

4. $[t_1] = [t_2]$

   pela definição das classes de equivalência $[t] = \{t' \mid t \equiv t'\}$.

5. $t_1 \equiv t_2$

   pela definição de congruência.

6. $\mathrm{val}(t_1) = \mathrm{val}(t_2)$, para todas as Esp-álgebras A

   para todo o $X = \emptyset$, $\overline{\mathrm{atr}} = \mathrm{val}$.

7. $e = (t_1, t_2)$, é válida em todas as Esp-álgebras A,

   para todo o $X = \emptyset$, tem-se $\overline{\mathrm{atr}} = \mathrm{val}$, e como tal $\mathrm{val}(t_1) = \mathrm{val}(t_2)$ ou seja e é válida em A.

III) É necessário mostrar que para toda a equação $(E, D) \in Eq$ e toda a atribuição $\text{atr}_x : X \to T_{\text{Esp}}$ temos que:

$$\overline{\text{atr}}_x(E) \quad = \quad \overline{\text{atr}}_x(D)$$

Se E e D são termos fechados então isto segue directamente da propriedade II. Caso contrário, e dado que val é sobrejectiva podemos definir uma atribuição $\text{atr} : X \to T_{OP}$ com

$$\text{atrx} \quad = \quad \text{val} \circ \text{atr}$$

Tomando $t_1 = \overline{\text{atr}}(E)$ e $t_2 = \overline{\text{atr}}(D)$ obtêm-se um par $e = (t_1, t_2)$ de termos fechados para os quais é necessário mostrar que satisfazem

$$\text{val}(t_1) \quad = \quad \text{val}(t_2) \quad \text{para toda a Esp-álgebra } A$$

Para tal constrói-se a atribuição $\text{atr}_A = \text{val}_A \circ \text{atr}$, usando a compatibilidade das atribuições tem-se

$$\text{val}_A(t_1) \quad = \quad \text{val}_A \circ \overline{\text{atr}}(E) = \overline{\text{atr}}_A(E)$$

e

$$\text{val}_A(t_2) \quad = \quad \text{val}_A \circ \overline{\text{atr}}(E) = \overline{\text{atr}}_A(D)$$

Dado que A é uma Esp-álgebra e $(E, D) \in Eq$ então $\overline{\text{atr}}_A(E) = \overline{\text{atr}}_A(D)$ ou seja

$$\text{val}_A(t_1) \quad = \quad \text{val}_A(t_2) \quad \text{para toda a Esp-álgebra}$$

mas então (pela propriedade II)

$$\text{val}(t_1) \quad = \quad \text{val}(t_2)$$

o que significa que

$$\text{val} \circ \overline{\text{atr}}(E) \quad = \quad \text{val} \circ \overline{\text{atr}}(D)$$

mas então por $\text{atrx} = \text{val} \circ \text{atr}$ e pela compatibilidade das atribuições tem-se

$$\overline{\text{atr}}_x(E) \quad = \quad \overline{\text{atr}}_x(D)$$

c.q.d.