AN AXIOMATIZATION OF THE RATIONAL DATA OBJECTS

J.A. Bergstra, A. Ollongren, Th.P. van der Weide
Institute of Applied Mathematics and Computer Science
University of Leiden, The Netherlands

0. Introduction

In the present paper we introduce the notion of a data-type system consisting of a many-sorted structure together with a finite set of axioms. This notion is a generalization of J.V. Guttag's [1] notion of abstract data-types. We distinguish between internal and external objects and we get the opportunity to split up a programming task into data-type implementation and data-type use. The language for the external objects is the one available to the user of a data-type system. The internal language can be used to introduce operations for instance in order to enable one to obtain more concise formulations of properties of data objects.

In our definition it is not necessary that the objects of the different sorts are finite, even though we will in practice mostly be interested in data objects which are finitely representable in some way.

The outline of the paper is as follows. After having introduced the notion of data-type systems in section 1, we define in section 2 the so-called rational objects (see H.D. Ehrich [2]), together with operations on them. Then, in section 3, it is shown how the rational data objects can be described as a data-type system. Also we give a standard model for rational data objects and formulate a few induction schemes which are useful for proving theorems on the class.

1. Data-type systems

A data-type system will consist of

i) a many-sorted structure A with a finite number of sets of different sorts($\vec{A}$), operations($\vec{F}$), relations($\vec{R}$) and constants($\vec{C}$) which are either internal or external;

ii) a finite set A of axioms in L^{int} (the first-order language of the structure), which all must be satisfied by the structure A.

We write for the many-sorted structure

$$A = \langle(\vec{A}^{int},\vec{A}^{ext}),(\vec{F}^{int},\vec{F}^{ext}),(\vec{R}^{int},\vec{R}^{ext}),(\vec{C}^{int},\vec{C}^{ext})\rangle$$

in which an arrow denotes the fact that we have a finite sequence.

Let N_i be sets of names for objects of external sort i.

We define two languages $L^{int}(\vec{N})$ and $L^{ext}(\vec{N})$ where the internal language L^{int} will be the first-order language over A and the external language L^{ext} contains only those formulas of the internal language satisfying the following constraints: only external operations, relations and constants are present and no free variables or quantifiers occur.

We will consider the following questions:

a) How can a data-type system be used?

b) What is a correct implementation of a data-type system?

As to a): we suppose that the following kind of actions are possible when working with a data-type system using objects in A_i :

- the assignment $n := c$, where n is a name and c is a constant compatible with n ;
- the assignment $n := F(—)$, where n is a name and F is an external function and $—$ is a sequence of names such that $F(—)$ is defined (i.e. the occurring names in $—$ have been assigned to) and is compatible with n .

In addition to these actions elements of the external language (i.e. $R(—)$ for an external relation defined for the sequence of names $—$) can be used as Boolean expressions.

As to b): we define as follows:

Definition 1.3.1.: A data-type implementation (DTI) for A is a semantic interpretation of the actions described above.

A DTI is correct for A and the set of axioms A if the following holds:

for all formula's $\phi \in L^{ext}(\vec{N})$: $A \vdash \phi \Rightarrow DTI \models \phi$

2. Rational data objects

Let Σ be a (finite or countable) set of selectors. E is a countable set of elementary data objects and $\omega \in E$ is called the null object.

Let $\mathcal{O} = \{F | F : \Sigma^* \to E\}$.

Let $\cdot$ be an operation: $\mathcal{O} \times \Sigma^* \to \mathcal{O}$ defined by: $A \cdot \sigma = \lambda\tau \cdot A(\sigma * \tau)$.

Further Ω is a constant given by: $\Omega = \lambda\sigma \cdot \omega$.

The domain Dom(A) of A is: $\{\sigma | A \cdot \sigma \neq \Omega\}$.

Now the rational data objects over Σ and E are defined as follows:

$$R_E^\Sigma = \{A \in \mathcal{O} | \exists_{S \subseteq \Sigma} [S \text{ finite} \wedge \mathrm{Dom}(A) \subset S^*] \wedge \{A \cdot \sigma | \sigma \in \Sigma^*\} \text{ finite}\}$$

2.1. Some important operations on rational objects

- ν operation

  for $A \in R_E^\Sigma$, $\sigma \in \Sigma^*$ and $e \in E$, the object $\nu(A,\sigma,e)$ is defined by

  $\nu(A,\sigma,e)(\tau) = \underline{\text{if}}\ \sigma = \tau\ \underline{\text{then}}\ e\ \underline{\text{else}}\ A(\tau)$

  From Ω, Σ^* and E we can build all finite objects using ν . (See [3] for a first order axiomatisation of a generalised class of finite objects which is based on ν).

- μ operation

  For $\sigma \in \Sigma^*$, A, B $\in R_E^\Sigma$ the object $\mu(A,\sigma,B)$ is defined by

  $\mu(A,\sigma,B)(\tau) = \underline{\text{if}}\ \tau = \sigma * \delta\ \underline{\text{then}}\ B(\delta)\ \underline{\text{else}}\ A(\tau)$

  It is easy to see that $\mu(A,\sigma,B) \in R_E^\Sigma$. μ plays an important role in the theory of Viennese data objects. See [3].

- ϕ operation

  For A, σ, e the object $\phi(A,\sigma,e) \in R_E^\Sigma$ is defined as follows:

<u>if</u> $\exists_{\tau\leq\sigma}[A(\tau) = e]$ <u>then</u> $\phi(A,\sigma,e) = \Omega$ <u>else</u> $\phi(A,\sigma,e)(\tau) =$ if δ is the shortest initial segment of τ s.t. $A(\delta) = e$ and $\tau = \delta * \tau'$ then $\phi(A,\sigma,e)(\sigma*\tau')$, if for no initial segment δ of τ $A(\delta) = e$ then $A(\tau)$.

We will go into the details of ϕ in section 3.

<u>Remark</u>: A rational object is in fact an equivalence class of Moore automata. The ϕ operation is easily understandable as a transformation of Moore automata.

- <u>θ transformations</u>

$T_\theta = \lambda A\cdot\lambda\sigma\cdot A(\theta(\sigma))$

in which θ is a finite transducer. $T_\theta(A)$ is again rational.

2.2. Operations of the form $A \to \lambda\sigma\cdot A(\theta(\sigma))$

Let f be a surjective function: $E \to \Sigma^*$

We define $M(A,\sigma)$ by: $\begin{cases} M(A,\varepsilon) = f(A(\varepsilon)) \\ M(A,\sigma*s) = M(A,\sigma) * f(A(\sigma*s)) \end{cases}$

<u>Definition 2.4.1.:</u> $\theta \in FT$ if it can be computed by a finite transducer.

<u>Theorem 2.4.1.:</u> $\theta:\Sigma^* \to \Sigma^* \in FT$ if and only if for some $A \in R_E^\Sigma : \theta = \lambda\sigma\cdot M(A,\sigma)$.

<u>Proof</u> easy. ⊠

In fact one can show that there are uncountably many $\theta:\Sigma^* \to \Sigma^*$ such that $\lambda\sigma\cdot A(\theta(\sigma)) \in R$ for all $A \in R$. A useful class of such θ might be GFT as defined below.

<u>Definition 2.4.2.:</u> GFT is the smallest class of functions $(:\Sigma^*\to\Sigma^*)$ containing FT and closed under composition and concatenation.

3. Rational objects as a data-type system

In defining a datatype system one has to decide which sets, operations, relations and constants are to be internal and which external. The decision taken should depend upon the problem to be studied in the system. In the present section we choose everything external for the sake of simplicity.

A finite set of axioms for the rational objects introduced before (section 2) together with some induction schemes are given. The operations we need: selector composition $*$, selection $\cdot$, ν , construction ϕ and application, are already known. If A is an object and x a selector path we write $A\cdot x$ for selection and $A(x)$ for application as before.

We will introduce induction schemes. However, only a finite number of occurrences of the inductions in the induction schemes are needed for proving important properties of rational objects. We do not take the trouble here to replace the induction schemes by occurrences.

<u>3.1.</u> We consider the following data-type system for rational objects

$DTS_{RO} = \langle(E,CS,\mathcal{O}), (*,\cdot,\nu,\phi), (\omega,\varepsilon,\Omega)\rangle$

No relations are introduced into the system at this moment. The type of ω is a constant in E ; $\mathcal{O}$ contains operations from CS to E . DTS_{RO} will be described in a language L and we require DTS_{RO} to satisfy a finite collection of axioms. CS is a set of selection paths and we first introduce a relation $<$ on $CS \times CS$ and

a predicate At on CS by the following definition.

Definition 3.1.1.: $y < x \iff \exists_\sigma[\sigma\neq\varepsilon \wedge y*\sigma=x]$ (y is prefix of x)

$At(x) \iff x \neq \varepsilon \wedge \forall_y[y<x \Rightarrow y=\varepsilon]$ (x is a selector)

Concerning CS we include the following axioms A0 :

- selection paths are sequences of simple selectors
- associativity
- the constant ε denotes the empty sequence
- scratching properties
- induction on the length of selection paths (sequence induction)
- there are at least two simple selectors.

Other axioms are:

A1 (sufficiently many elementary objects) $\exists_e\forall_x[A(x) \neq e]$

A2 (selection and application) $A\cdot(\alpha*\beta) = (A\cdot\alpha)\cdot\beta \wedge A\cdot\varepsilon = A$

A3 (empty object) $\Omega(\alpha) = \omega$

A4 (extensionality) $\forall_\alpha[A(\alpha) = B(\alpha)] \Rightarrow A = B$

Lemma 3.1.1.: $R_E^\Sigma \models$ A0 - A4

Proof: omitted.

The axioms for the ϕ - and the ν operation are:

A5 $\phi(A,\rho,m)(\alpha) =$ if $\alpha = \beta * \gamma \wedge A(\beta) = m \wedge \forall_{\sigma<\beta}[A(\sigma)\neq m]$ then $\phi(A,\rho,m)(\rho*\gamma)$ else $A(\alpha)$

(Note that β and γ are uniquely determined)

A6 $\nu(A,\alpha,e)(\beta) =$ if $\alpha = \beta$ then e else $A(\beta)$

In the next section we shall outline how each rational object can be built using only ν - and ϕ operations, starting from Ω . We express this fact in the following axiom scheme of construction induction.

A7 $\Phi(\Omega,\text{—})$

$$\left.\begin{array}{l} \wedge\ \Phi(A,\text{—}) \Rightarrow \forall_{\alpha,e}\ \Phi(\nu(A,\alpha,e),\text{—}) \\ \wedge\ \Phi(A,\text{—}) \Rightarrow \forall_{\rho,m}\ \Phi(\phi(A,\rho,m),\text{—}) \end{array}\right\} \Rightarrow \forall_A[\Phi(A,\text{—})]$$

(Here — denotes an unspecified list of arguments not containing the first argument of Φ).

The theory of A0 - A7 is called T_{RO} and it is described in a language L .

3.2. We shall introduce a special class of Moore automata (MA) and a class of generalised Moore automata (GMA), with the following purpose in mind: we intend to show that each rational object can be built in a finite number of steps (ν and ϕ). As we wish to have this mechanism in the theory for other reasons, we develop the notions in the theory itself.

First we introduce list objects. They are used for encoding finite ordered sets of information (= objects). We will call L a list object if

- $L(\alpha) \neq \omega \Rightarrow \exists_\delta[\alpha=\delta*\gamma\wedge\delta\in\{s\}^**t]$
- for some $s^{n+1}\in\{s\}^*$ $L\cdot s^{n+1} = \Omega$

We say: L encodes the objects $L\cdot(s^i*t)$ $i=1,2,\ldots,n$

Remarks: Even though the natural numbers are not explicitly available in the theory,

induction on the length of list objects is available (sequence induction).

The predicate $\sigma \in \{s\}^*$ is expressible in the theory as follows:

$\sigma \in \{s\}^* \iff \forall_{\alpha,\beta,\gamma}[\alpha*\beta*\gamma=\sigma \wedge At(\beta) \Rightarrow \beta=s]$

Now we are able to introduce generalised Moore automata. They are used for combining two views on rational objects: (1) the extensional one (a rational object seen as a function from Σ^* to E) and (2) a rational object seen as a Moore automaton, in order to be able to describe operations on rational objects as operations on automata.

We call B a GMA if:

- $B(\varepsilon) = \omega$
- $B\cdot s$ is a list object, encoding pairs consisting of a marking symbol m , and a selection-path ρ , in such a way that: $s^i \in Dom(B\cdot s) \Rightarrow \exists!_\rho[B\cdot s(s^i*t*\rho)=m\neq\omega]$

  $B\cdot s(\alpha) = B\cdot s(\beta) \Rightarrow \alpha = \beta$
- $B\cdot t$ is an object such that:

  $B\cdot t(\alpha)$ is a marking symbol $\Rightarrow \forall_{\beta\neq\varepsilon}[B\cdot t(\alpha*\beta)=\omega]$

  $B\cdot t(\alpha)$ is a marking symbol $\Rightarrow \forall_{\gamma<\alpha}[B\cdot t(\gamma)$ is not a marking symbol$]$

  $B\cdot t(\alpha)$ is a marking symbol, coupled with selectionpath $\rho \Rightarrow \rho < \alpha$

We introduce: $TR(B) = B\cdot t$

$RL(B) = B\cdot s$

When B is a GMA , the corresponding rational object (denoted by $\tilde{B}$) can be found, by unfolding $TR(B)$, using the links encoded in $RL(B)$. We define $\tilde{B}$ by:

$\tilde{B}(\sigma) =$ <u>if</u> $\sigma = \alpha * \beta \wedge TR(B)(\alpha)$ is a marking symbol coupled with ρ

<u>then</u> $\tilde{B}(\rho*\beta)$ <u>else</u> $TR(B)(\sigma)$.

Using a list object, encoding the computation of the GMA B on argument σ we can express: $\tilde{B}(\sigma) \equiv e$.

And therefore, we also can express: $A \equiv \tilde{B}$.

We call B a MA , if it is a finite GMA . This can be expressed using list objects.

We show some important properties:

<u>Lemma 3.2.1.:</u> $R_E^\Sigma \models \forall_A \exists_{B\in MA}[A\equiv\tilde{B}]$

<u>Proof:</u> Let n be the number of selection results of A and let $s_1,s_2,\ldots,s_m$ be a set of selectors such that $Dom(A) \subseteq \{s_1,s_2,\ldots,s_m\}^*$. We construct a GMA B as follows. We choose n^m elementary objects not occurring in A : for each $\sigma \in \{s_1,s_2,\ldots,s_m\}^n$ a new elementary object m_σ is chosen. Let ρ_σ be the shortest initial segment of σ such that $A\cdot\rho_\sigma = A\cdot\sigma$. B is then defined by $TR(B)$ and $RL(B)$ where

$$TR(B)(\sigma) = \begin{cases} A(\sigma) & \text{if } length(\sigma) < n \\ m_\sigma & \text{if } length(\sigma) = n \\ \omega & \text{otherwise} \end{cases}$$

$RL(B)$ is a list consisting of all pairs (ρ_σ,m_σ)

With induction on the length of τ one proves

$\forall_\tau \forall_{\sigma\in Dom(TR(B)),TR(B)(\sigma)\neq m}[\tilde{B}(\sigma*\tau)\equiv A(\sigma*\tau)]$ ∎

Lemma 3.2.2.: $R_E^{\Sigma} \models \forall_{B \in GMA} \exists_A [A \equiv \tilde{B}]$

Proof: Induction on the length of $RL(B)$.

If $RL(B) = \Omega$ then $\tilde{B} = TR(B)$

If $RL(B)$ has the form $\nu(L, \sigma * s * t * \rho, m)$, where $\sigma \in Dom(L)$ and $\sigma * s \notin Dom(L)$, then we define the GMA B' as follows:

$RL(B') = L$ and $TR(B') = \phi(TR(B), \rho, m)$

Without proof we state that $\tilde{B}' \equiv \tilde{B}$ and that B' is again a GMA ⊠

Lemma 3.2.3.: Every rational object can be built by the application of a finite number of ν - and ϕ operations starting from Ω .

Proof: Let A be a rational object and let B be an MA such that $\tilde{B} \equiv A$ (lemma 3.2.1.). First construct $TR(B)$. After a finite number of applications of ϕ in the way outlined in the proof of lemma 3.2.2. the result is the rational object A itself. ⊠

Theorem 3.2.1.: $R_E^{\Sigma} \models$ A0 — A7

Proof: immediate with the preceding lemma's. ⊠

3.3. Now we look at what can be proved inside the theory. The following facts which can be expressed in L , can be proved easily using construction induction.

i) For each A there exists a list object L which codes $\{s_1, s_2, \ldots, s_n\}$ such that $Dom(A) \subseteq \{s_1, s_2, \ldots, s_n\}^*$

ii) For each A there exists a list object L which codes the set of selection results of A .

By formalizing the arguments from the previous section we can establish

Theorem 3.3.1.: $T_{RO} \vdash \forall_A \exists_{B \in MA} [A \equiv \tilde{B}]$

$T_{RO} \vdash \forall_{B \in MA} \exists_A [A \equiv B]$

Remarks: As a consequence of this theorem a coding of the rational objects in finite ones is available in the theory. This enables us, for example, to give formal existence proofs of $\mu(A, \sigma, B)$ and $\lambda\sigma \cdot A(\sigma * \sigma)$ for arbitrary A and B .

References

[1] J.V. Guttag 1975, "The specification and Application to Programming of Abstract Data Types", Technical Report CSRG-59, Dept. of Electrical Engineering and Dept. of Computer Science, University of Toronto.

[2] H.D. Ehrich 1976, "Outline of an algebraic theory of structured objects", Automata, Languages and Programming, Third International Colloquium, Edinburgh University Press, p. 508-530.

[3] G.A. Terpstra, Th.P. van der Weide, H.J.M. Goeman, A. Ollongren, J.A. Bergstra 1976, "Uniform axioms for structured objects", Report No. 76-7, Institute of Applied Mathematics and Computer Science, University of Leiden.

[4] J.A. Bergstra, A. Ollongren, Th.P. van der Weide, "An axiomatization of the rational data objects", Report No. 77-2, Institute of Applied Mathematics and Computer Science, University of Leiden.