A CHARACTERIZATION OF ABSTRACT DATA AS MODEL-THEORETIC INVARIANTS
A. Bertoni - G. Mauri - P.A. Miglioli
Istituto di Cibernetica - Università di Milano

1. INTRODUCTION

The problem of abstract data specification has been seriously taken into account in a lot of recent papers, and one of the most accredited approaches is the algebraic one, as developed by Liskov and Zilles [9], Zilles [15], Outtag [5] and ADJ [4], that hinges on the following theses:

a) a data type is a many-sorted equational algebra;

b) an abstract data type is an isomorphism class of initial many-sorted equational algebras.

This approach gives rise to various difficulties, as pointed out by Majster [11], Klaeren [7], ADJ [14] and by the authors in [1]. In particular, the authors believe that the improvements proposed by ADJ [14] are not sufficient to overcome all the technical difficulties connected with the initial algebra approach, and in [1] they proposed, by means of some examples, a more general approach based on model-theoretic concepts and techniques. In this frame, not only equational axioms are to be taken into account in order to specify abstract data types, but the full expressive power of first order languages can be conveniently used. Furthermore, the model-theoretic point of view has clearly shown the need of requiring something more than initiality: the main thesis of [1] is that an abstract data type is an isomorphism class of models of a first order theory which are at the same time initial and prime.

The present paper is a further development of [1]: here we want not only to work out our ideas by examples and theses, but to revise, in a model theoretic frame, the notion itself of "abstract datum" and, on the basis of an intuitive analysis of this concept, to provide an adequate formalization of it.

Our starting point is any first order theory $T$ of general kind (i.e. a set of first order sentences, not necessarily equivalent to a set of universally quantified equations), where, in order to simplify the treatment, we require that the language of $T$ contains only functional symbols (together with, of course, the relational symbol $=$).

In this frame, an abstract datum on $T$ can be defined starting from a particular formula $\Delta$ provable from $T$, in such a way that, in every model $\mathcal{M}$ of $T$, there is a unique element (the concrete datum) satisfying the formula $\Delta$. Here the difference between the concrete datum and the abstract datum (independent from any model of $T$) is that the latter turns out to be the formula $\Delta$ itself (to be more precise, an appropriate equivalence class of formulas to which $\Delta$ belongs), that defines a model-theoretic invariant in the sense of Kreisel [8].

As a consequence, the notion of abstract data type follows in a natural way from our definition of abstract datum: under appropriate requirements, the set of all the abstract data can be structured as an algebra $\mathcal{A}$; when this algebra turns out to be a model of $T$, we say that $T$ admits an abstract data type, and call $\mathcal{A}$ the abstract data type on $T$.

A strong semantical characterization of the theories which admit abstract
data types is then provided. As a consequence of a theorem of Kreisel [8], we can prove that a theory T admits an abstract data type if and only if there is a model \( \mathcal{A} \) of T such that, for every model \( \mathcal{M} \) of T, there is a unique monomorphism from \( \mathcal{A} \) to \( \mathcal{M} \): we will refer to this property as monoinitiality of \( \mathcal{A} \), as opposed to initiality, that requires the existence of a unique morphism, mono or not.

We remark that monoinitiality captures abstractness for data types just as initiality; furthermore, we show that monoinitiality is a weaker property than initiality plus primeness, but it is independent (i.e. there are no implications) of initiality: so, our approach, based on monoinitiality, is essentially different from the one of ADJ [4].

As a second result, we are able to show that if a theory T is recursively axiomatizable and admits an abstract data type \( \mathcal{A} \), then there is \( \mathcal{A}' \neq \mathcal{A} \) whose defining operations are recursive: so an essential adequacy requisite for an abstract data specification technique, i.e. to capture recursiveness, is fulfilled.

Finally, we point out that the concept of monoinitiality often critically depends on the presence in the theory of axioms with inequalities; an example presented by Oppen [12] shows that the explicit assignment of such negative axioms in the theory of LISP list structures leads to efficient decision algorithms.

2. FUNDAMENTAL DEFINITIONS

The basic notions we need are those of relational structure (generalizing that of algebra) and of first order language.

We start with a many-sorted alphabet \( \mathbb{A} \), that consist of:
a) a set \( S \) of sorts;
b) a set \( \Sigma \) of operation symbols, together with an arity function \( \nu: \Sigma \rightarrow S^+ x S \);
c) a set \( R \) of relation symbols, together with an arity function \( \nu: R \rightarrow S^+ \);
d) a set \( C \) of constant symbols, together with a sort function \( \nu: C \rightarrow S \).

Def. 2.1 - A structure for \( \mathbb{A} \) is a pair \( \mathcal{M} = \langle M, i_M \rangle \) where \( M = \{ M_s | s \in S \} \) is a class of non empty sets indexed by \( S \), called the carriers of the structure, and \( i_M \) is the interpretation function, that associates:

i) with every \( \sigma \in \Sigma \) such that \( \nu(\sigma) = \langle s_1, \ldots, s_k \rangle \), a function \( i_M(\sigma) = \sigma: M_{s_1} \times \ldots \times M_{s_k} \rightarrow M_{s} \);

ii) with every \( r \in R \) such that \( \nu(r) = \langle s_1, \ldots, s_k \rangle \), a relation \( i_M(r) = \{ m \in M_{s_1} \times \ldots \times M_{s_k} \} \).