ABSTRACT: This paper presents a method of proof inspired from the works of Musser, Goguen, Huet and Hullot. The method of proof described here is more general and requires simpler, less constraining hypotheses. As a matter of fact, a specification of an equational theory would be said "fair" if it can be structured into smaller, one-sorted presentations, each of them partitioned in two: the first part expresses the relations between the data type generators, the second one can be formed into a canonical term rewriting system. Thus "fairness" extends the sufficient conditions given by Huet and Hullot for deciding what they call "the Definition Principle". Moreover, "fairness" is a very easy to respect hypothesis, in so far as it only consists in syntactical conditions. However our method requires explicitly the invocation of an inductive rule of inference, but we show how heuristics can be chosen accordingly in order to gain full advantage from our framework. Finally we outline how this method can be extended in order to automatically transform a given "fair" presentation into another equivalent one.

INTRODUCTION: There has been recently a growing interest in algebraic methods for deciding the equivalence of expressions by applying rewrite rules, and for proving inductive equational hypotheses without explicit invocation of an inductive rule of inference [MUS 80], [COG 80], [H & H 80]. All these methods use the fact that under certain conditions, an equation is true if and only if it is consistent. Further assumptions allow to relate consistency to the Knuth-Bendix algorithm, thus allowing to prove inductive properties without requiring explicit invocation of an inductive rule of inference. Our claim is that the validity problem cannot always be reduced to a consistency problem in practice. In this paper we therefore describe a method of proof which is, though perhaps less efficient, more general. However we shall explicitly use an inductive rule of inference. As all the authors referred to above did, we shall study the validity problem in the algebraic framework of abstract data types. Note that we are not interested here in "varietal semantics" but in "initial algebra semantics", i.e. we wonder if some equation holds in the standard initial model of an equational variety. The plan of this paper is as follows. Section 2 provides the basic definitions and results we shall use. In Section 3 we define the central notion of "fair presentation": fairness consists in syntactical conditions which are easy to respect but are sufficient to ensure the soundness of our method of proof. In Section 4 we briefly compare the methods of proof proposed by Musser, Goguen and Huet and Hullot. In Section 5 we describe our method of proof. This method is illustrated on a small, tutorial-level example. In the last section we show how the method of proof described in this paper can be used for the automatic transformation of a given fair presentation into another equivalent one.

II. BACKGROUND: This section briefly introduces basic definitions and results related to the algebraic theory of abstract data types. However, as we shall concentrate on the validity problem, we don't treat in detail here some peculiarities of our personal approach; they are described in [BID 81], [BID 82a]. Moreover we assume familiarity with the terminology of term rewriting systems as well as with that of many-sorted algebras. Most of the results described in this section are detailed and proved in [McL 71], [ADJ 78,79], [H & O 80] and [COG 80]. Given a set of sorts S and a signature Σ over S, we denote by Alg(S) the category which objects are all Σ-algebras and which arrows are all the Σ-morphisms between these algebras. This category has an initial object G, the Σ-algebra of ground terms over Σ. Another Σ-algebra of special interest is the Σ-algebra freely generated by a set of variables V, denoted by Tc(V) or T(Σ) for short. A Σ-equation is a pair of terms (with variables) M, N of same type (i.e. both M and N belong to a same carrier T(Σ)) ; equations are written M = N. A Σ-algebra A is a model of some equation M = N if we have A(M) = A(N) for every assignment v:V → A we have v(M) = v(N).
The equational variety of all models of some set $E$ of equations is denoted by $\text{ALG}(\Sigma,E)$. It is a full subcategory of $\text{ALG}(\Sigma)$ and it has also an initial object. The equational theory $\Sigma_E$ defined by a set $E$ of equations is the least $\Sigma$-congruence over $T(\Sigma)$ generated by $E$. The initial model is by definition the quotient algebra of the $\Sigma$-algebra $G$ of ground terms by the equational theory $\Sigma_E$. This initial model, denoted by $G(E)$ or $G(\Sigma,E)$ is a $\Sigma$-algebra model of $E$ and it is initial in the category $\text{ALG}(\Sigma,E)$. A $\Sigma$-algebra $C$ is called a canonical algebra if and only if:

- all the carrier of $C$ are formed with ground terms ($C_s \subseteq G_s$)
- if $Ft_1 \ldots t_n$ is a ground term of $C$, then all subterms $t_i$ belong to $C$ and $C_F(t_1, \ldots, t_n) = Ft_1 \ldots t_n$.

For a given set $E$ of equations, there always exists a canonical algebra initial in the category $\text{ALG}(\Sigma,E)$ and therefore isomorphic to $G(\Sigma,E)$. We provide now the definitions of abstract data type specification and presentation. We start with:

**Definition 1**: A specification is a triple $\langle S, \Sigma, E \rangle$ where $S$ is a set of sorts, $\Sigma$ a signature over $S$ and $E$ a set of $\Sigma$-equations. The abstract data type specified by $\langle S, \Sigma, E \rangle$ is the initial object (defined up to isomorphism) of the category $\text{ALG}(\Sigma,E)$. Two specifications $\langle S_1, \Sigma_1, E_1 \rangle$ and $\langle S_2, \Sigma_2, E_2 \rangle$ are equivalent if and only if they specify the same abstract data type, i.e.:

- $S_1 = S_2$ and $\Sigma_1 = \Sigma_2$
- $G(E_1) \models E_2$ and $G(E_2) \models E_1$

We now propose the following working definition for a "type of interest presentation".

**Definition 2**: Let $\langle S, \Sigma, E \rangle$ be some abstract data type specification.

A presentation (with respect to specification $\langle S, \Sigma, E \rangle$) is a triple $\langle T, \Sigma, E \rangle$ such that:

- $T$ is a new sort ($T \notin S$)
- $\Sigma$ is a signature over $S' = S \cup \{T\}$ verifying:
  * $\Sigma \cap \Sigma' = \emptyset$
  * For each operator $F$ in $\Sigma$, there is at least one domain or codomain of $F$ which is the type of interest $T$.
- $E$ is a set of $\Sigma'$-equations ($\Sigma' = \Sigma \cup \Sigma$) such that, for each equation $e$ in $E$, one operator from $\Sigma$ at least occurs in $e$.

Two presentations $\langle T_1, \Sigma_1, E_1 \rangle$ and $\langle T_2, \Sigma_2, E_2 \rangle$ (with respect to the same context specification $\langle S, \Sigma, E \rangle$) are equivalent if and only if they present the same type of interest, i.e.:

- $T_1 = T_2$ and $\Sigma_1 = \Sigma_2$
- $G(\Sigma', E \cup E_1) \models E_2$ and $G(\Sigma', E \cup E_2) \models E_1$ where $\Sigma'$ denote $\Sigma \cup \Sigma$.

There are some remarks we wish to make about the above definition of presentation.

First, note that requiring these syntactical conditions assures us that the triple $\langle T, \Sigma, E \rangle$ is a specification. Moreover, as well as abstract data types are interpreted as (initial) $\Sigma$-algebras, the type of interest presented by $\langle T_1, \Sigma_1, E_1 \rangle$ can be interpreted as a functor $T_1 : \text{ALG}(\Sigma) \to \text{ALG}(\Sigma')$ left-adjoint to the forgetful functor $U : \text{ALG}(\Sigma') \to \text{ALG}(\Sigma)$.

The interest of such a framework, which provides a rigorous formalization of data types extensions, as well as leading to concise and elegant characterizations of consistency and sufficient completeness, is emphasized in [BID 82a]. Briefly, the abstract data type specified by $\langle S', \Sigma', E' \rangle$ is actually the abstract data type specified by $\langle S, \Sigma, E \rangle$ (the "context" of the type of interest) enriched by the type of interest presented by $\langle T_1, \Sigma_1, E_1 \rangle$. Furthermore, the following definition states at which condition an abstract data type could be considered as the enrichment of another one: