An effective theorem proving algorithm

By P. ECSEDI—TÓTH and A. VARGA

Summary

A simple procedure is presented which is a sound and complete calculus for first-order logic. The method can easily be implemented on computers and it has considerable advantage in manual work.

Introduction

Nowadays computer science becomes more and more 'scientific' after its 'naive' age of the last 30 years. Much work has been devoted to create 'software industry', however, for although there is a shortage of really effective methods.

Research of artificial intelligence gives rise to a hope of solving this problem. First program-correctness-proving procedures were suggested by Turing, Floyd, Manna and others in their works on 'modeling' intelligent processes of human thinking. The soul of these procedures is a pure mathematical logical tool: a theorem-proving algorithm, or in other words an automatic deduction.

After the activity of Gödel, Skolem, Turing, and Herbrand [5, 9; 10, 6], J. A. Robinson proposed an elegant practical realization of automatic deduction in [8]. Since 1965, when the first paper on Robinson's resolution principle appeared, many theorem-proving algorithms, each of them is an 'improvement' of the resolution principle, have been published. However, almost all these methods have two common disadvantages. Namely, they have a deep combinatorial nature and one has to do a not-so-few unnecessary steps in pre-procedure.

The input of a theorem-prover, working on the basis of a version of the resolution principle, is a formula in the first-order logic. (First-order logic is one of the most important tools for 'software industry', but theorem-provers of other types of languages e.g. zero- or second-order logic, non-classical (modal) logic, fuzzy logic and other multi-valued logics etc. are also known.) This input formula has to be in prenex normal form, in disjunctive normal form and in Skolem-normal form. In this paper we present a procedure which can be applied to formulae not being in disjunctive normal form (skolemization and prenex normal form are needed).

Our method has some similarity, although it is essentially different from that, to the resolution principle. Our procedure is a 'valuation' or rather a 'computation of logical value' of the given formula, and thus it has less combinatorial character.

Some other resolution-like algorithms eliminating the steps needed to get a disjunctive normal form of the given input formula, are also known [7], [2, 3] but our method over the 'computation' property, mentioned above, differs in the following points, as well. It offers:

i) Easier treating of quantified variables;
ii) More effectiveness thanks to the less combinatorial nature of our method;
iii) Easier manual and also machine execution.

In the first section the notions, notations and facts needed for the development are surveyed.

The second section deals with the essence of the material. In the third one the procedure is defined for first-order logic. Completeness theorems are only stated, the long but not too complicated proofs will appear elsewhere.

§ 1. Preliminaries

By a type τ we mean an ordered quintuple

$$\tau = \langle I, J, K, t', t'' \rangle$$

where I, J, K are pairwise disjoint sets (sets of relation symbols, function symbols and constant symbols, respectively); t', t'' are functions for which

$$\begin{aligned} &\text{Domain } (t') = I, \\ &\text{Range } (t') \subset \omega, \\ &\text{Domain } (t'') = J, \\ &\text{Range } (t'') \subset \omega. \end{aligned}$$

The set of τ-type formulae of zero- and first-order (defined in the standard way) will be denoted by

$${}_0F^\tau \quad \text{and} \quad {}_1F^\tau.$$

$\mathfrak{S}^\tau$ is the class of τ-type algebraic structures (the class of possible models). Let $M(\varphi)$ denote the following class of structures

$$M(\varphi) = \{\mathfrak{A} \mid \mathfrak{A} \in \mathfrak{S}^\tau, \mathfrak{A} \models \varphi\}.$$

$\varphi \in {}_1F^\tau$ is a tautology (unsatisfiable) iff

$$M(\varphi) = \mathfrak{S}^\tau \quad (M(\varphi) = \emptyset).$$

The notion of algorithm is used in a naive sense. We shall suppose that a finite set of symbols Z is fixed.

Let

$$Z^* = \{\langle z_1, \ldots, z_n\rangle | n < \omega, z_i \in Z \quad \text{if} \quad i \leqq n\}.$$

By the definition of an algorithm we mean an element of Z^*.

If $z \in Z^*$ then there exists a partial function

$$\tilde{z}: Z^* \to Z^*,$$

the meaning of z. Consequently, if $z_1, z_2, z_3 \in Z^*$ then

$$\tilde{z}_1(z_2) = z_3$$

means that the result of applying the algorithm z_1 to the algorithm z_2 is the algorithm z_3.

We will assume that the elements of ${}_1F^\tau$ and ω are represented in Z^* (for example ${}_1F^\tau \subset Z^*$ and $\omega \subset Z^*$).

Let $z \in Z^*$ and $A \subset Z^*$. z enumerates the set A iff

$$\text{Domain } (\tilde{z}) = \omega,$$

$$\text{Range } (\tilde{z}) = A.$$

Suppose there exist two distinguished elements in Z^*, namely, z_0 and z_1 which represent the truth-values TRUE and FALSE, respectively.

If $z \in Z^*$, $A \subseteqq Z^*$ then z decides the set A iff

$$\text{Domain } (\tilde{z}) = Z^*,$$

$$\text{Range } (\tilde{z}) = \{z_0, z_1\}$$

and for any $z' \in Z^*$,

$$\tilde{z}(z') = z_0 \quad \text{iff} \quad z' \in A.$$

§ 2. Deciding the set of zero-order sentences

For an arbitrary zero-order sentence φ, let S_φ be a sequence of prime sentences (relation symbols with no variables in their argumentum places) occurring in φ.

Let φ be a zero-order sentence. The following algorithm z defines a binary tree $\tilde{z}(\varphi)$ of φ:

Step 1. Start with the sentence φ as initial vertex.

Step 2. Choose the first not used element of S_ψ, say p, where ψ is the actually treated vertex, and substitute the truth-values TRUE and FALSE, respectively, for p in ψ. Draw two edges, labelled by p and its negation, respectively, from the vertex.

Step 3. Apply the following rules to obtain the truth-value-free version o the substituted vertex ψ or a single truth-value along both of the edges: for any zero-order sentence φ,

$$\varphi\wedge 1 \leftrightarrow \varphi; \quad 1\wedge\varphi \leftrightarrow \varphi \tag{1}$$

$$\varphi\wedge 0 \leftrightarrow 0; \quad 0\wedge\varphi \leftrightarrow 0 \tag{2}$$

$$\varphi\vee 1 \leftrightarrow 1; \quad 1\vee\varphi \leftrightarrow 1 \tag{3}$$

$$\varphi\vee 0 \leftrightarrow \varphi; \quad 0\vee\varphi \leftrightarrow \varphi \tag{4}$$

$$\varphi \rightarrow 1 \leftrightarrow 1 \tag{5}$$

$$\varphi \rightarrow 0 \leftrightarrow \bar{\varphi} \tag{6}$$

$$1 \rightarrow \varphi \leftrightarrow \varphi \tag{7}$$

$$0 \rightarrow \varphi \leftrightarrow 1 \tag{8}$$

$$(\varphi \leftrightarrow 1) \leftrightarrow \varphi; \quad (1 \leftrightarrow \varphi) \leftrightarrow \varphi \tag{9}$$

$$(\varphi \leftrightarrow 0) \leftrightarrow \bar{\varphi}; \quad (0 \leftrightarrow \varphi) \leftrightarrow \bar{\varphi} \tag{10}$$

$$\varphi\wedge\varphi \leftrightarrow \varphi \tag{11}$$

$$\varphi\wedge\bar{\varphi} \leftrightarrow 0; \quad \bar{\varphi}\wedge\varphi \leftrightarrow 0 \tag{12}$$

$$\varphi\vee\varphi \leftrightarrow \varphi \tag{13}$$

$$\varphi\vee\bar{\varphi} \leftrightarrow 1; \quad \bar{\varphi}\vee\varphi \leftrightarrow 1 \tag{14}$$

$$\varphi \rightarrow \varphi \leftrightarrow 1 \tag{15}$$

$$(\varphi \leftrightarrow \varphi) \leftrightarrow 1 \tag{16}$$

$$\varphi \leftrightarrow \bar{\bar{\varphi}} \tag{17}$$

$$\bar{1} \leftrightarrow 0; \quad \bar{0} \leftrightarrow 1 \tag{18}$$

where $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $^{-}$, 0 and 1 is the symbol for conjunction, disjunction, implication, equivalence, negation, truth-values FALSE and TRUE, respectively.

The truth-value-free formulae or single truth-values, obtained from the actually treated vertex will be the two new vertices χ and η. Let the sequence of zero-order prime sentences of χ (η) be S_χ (S_η) with the same order as in S_φ.

Note that $S_\chi = S_\psi - \{p\}$ $(S_\eta = S_\psi - \{p\})$.

Step 4. Do steps 2 and 3 for the vertices χ and η if they are not single truth-values. If each of the vertices newly made is a truth-value, the procedure terminates.

The vertices which are truth-values will be called final-vertices.

Example 1. Assume that the given zero-order sentence is

$$\varphi = \overline{(A(a, b)\wedge\overline{C})} \to ((B(a) \vee C) \leftrightarrow A(a, b))$$

where

$$t'(A) = 2, \quad t'(B) = 1, \quad t'(C) = 0, \quad t''(a) = 0, \quad t''(b) = 0.$$

The tree produced by the algorithm is indicated in Fig. 1, and its "computation" in Fig. 2.

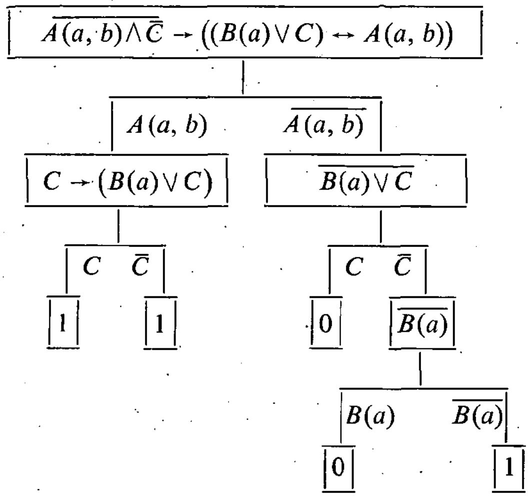

Fig. 1

Let ${}_0f^{\mathrm{dec}}$, ${}_0g^{\mathrm{dec}}$ be algorithms such that

i)
$$ {}_0\tilde{f}^{\mathrm{dec}}: {}_0F^{\tau} \to \{z_0, z_1\};$$
$$ {}_0\tilde{g}^{\mathrm{dec}}: {}_0F^{\tau} \to \{z_0, z_1\},$$

ii)
$$ {}_0\tilde{f}^{\mathrm{dec}}(\varphi) = \begin{cases} z_0 & \text{if all the final-vertices of } \tilde{z}(\varphi) \text{ are } 1 \\ z_1 & \text{otherwise,} \end{cases}$$

$$ {}_0\tilde{g}^{\mathrm{dec}}(\varphi) = \begin{cases} z_0 & \text{if all the final-vertices of } \tilde{z}(\varphi) \text{ are } 0 \\ z_1 & \text{otherwise.} \end{cases}$$

Note that ${}_0f^{\mathrm{dec}}$ and ${}_0g^{\mathrm{dec}}$ do exist, i.e., the final vertices of $\tilde{z}(\varphi)$ do not depend on the order of S_{φ}.

Theorem. [11].
i) ${}_0f^{\mathrm{dec}}$ decides the set of zero-order tautologies;
ii) ${}_0g^{\mathrm{dec}}$ decides the set of zero-order unsatisfiable sentences.

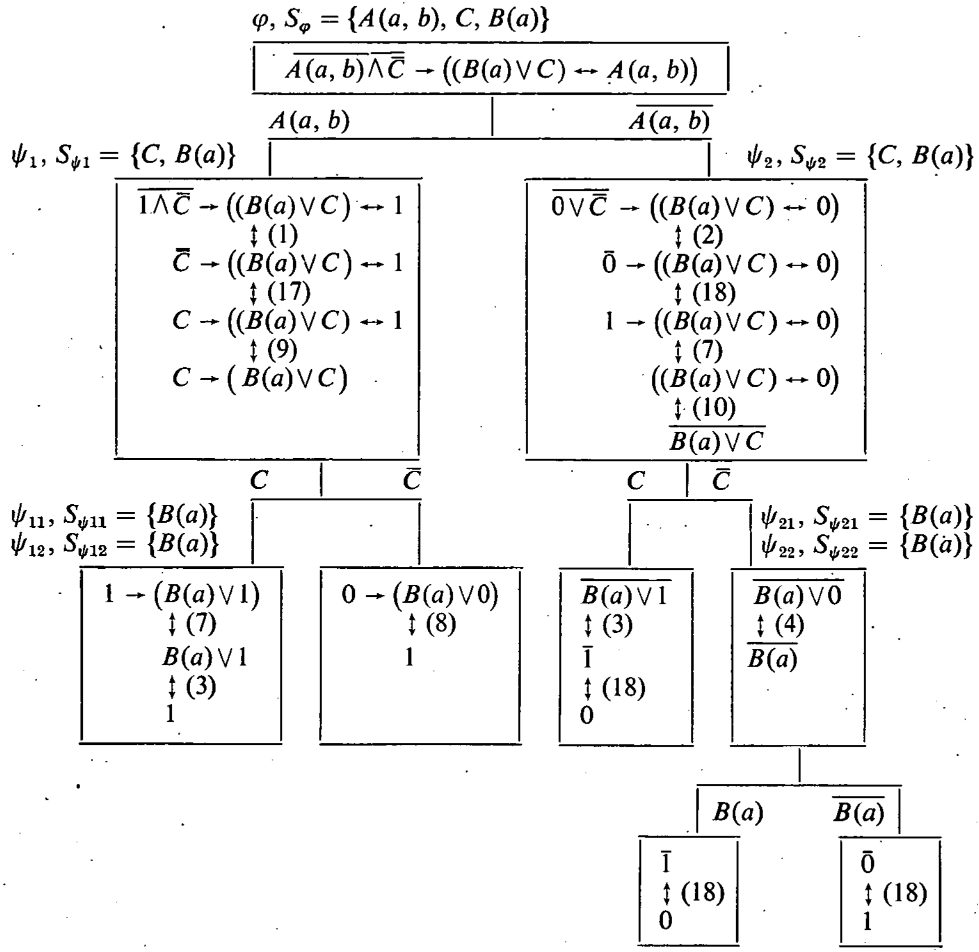
$\varphi, S_\varphi = \{A(a,b), C, B(a)\}$
$\overline{A(a,b)} \wedge \overline{C} \to ((B(a) \vee C) \leftrightarrow A(a,b))$
$A(a,b)$
$\overline{A(a,b)}$
$\psi_1, S_{\psi 1} = \{C, B(a)\}$
$\psi_2, S_{\psi 2} = \{C, B(a)\}$
$\overline{1} \wedge \overline{C} \to ((B(a) \vee C) \leftrightarrow 1$ ↕ (1) $\overline{C} \to ((B(a) \vee C) \leftrightarrow 1$ ↕ (17) $C \to ((B(a) \vee C) \leftrightarrow 1$ ↕ (9) $C \to (B(a) \vee C)$
$\overline{0} \vee \overline{C} \to ((B(a) \vee C) \leftrightarrow 0)$ ↕ (2) $\overline{0} \to ((B(a) \vee C) \leftrightarrow 0)$ ↕ (18) $1 \to ((B(a) \vee C) \leftrightarrow 0)$ ↕ (7) $((B(a) \vee C) \leftrightarrow 0)$ ↕ (10) $\overline{B(a) \vee C}$
C
$\overline{C}$
$\psi_{11}, S_{\psi 11} = \{B(a)\}$
$\psi_{12}, S_{\psi 12} = \{B(a)\}$
$\psi_{21}, S_{\psi 21} = \{B(a)\}$
$\psi_{22}, S_{\psi 22} = \{B(a)\}$
$1 \to (B(a) \vee 1)$ ↕ (7) $B(a) \vee 1$ ↕ (3) 1
$0 \to (B(a) \vee 0)$ ↕ (8) 1
$\overline{B(a) \vee 1}$ ↕ (3) $\overline{1}$ ↕ (18) 0
$\overline{B(a) \vee 0}$ ↕ (4) $\overline{B(a)}$
$B(a)$
$\overline{B(a)}$
$\overline{1}$ ↕ (18) 0
$\overline{0}$ ↕ (18) 1

Fig. 2

§ 3. Tree-constructing for first-order sentences

We define the well-known Skolem-extension of type τ in the following way:

$$\tau^{(0)} = \tau.$$

If

$$\tau^{(m)} = \langle I, J^{(m)}, K, t', t''^{(m)} \rangle \quad \text{is defined}$$

then

$$\tau^{(m+1)} = \langle I, J^{(m)} \cup J^*, K, t', t''^{(m+1)} \rangle$$

where

$$J^* = \{f_\varphi | \varphi(x_1, \ldots, x_n) \in_1 F^{\tau^{(m)}}\}$$

and f_φ is a new function symbol for every $\varphi(x_1, \ldots, x_n) \in {}_1F^{\tau^{(m)}}$ and

$$t''^{(m+1)}(f_\varphi) = n-1;$$

furthermore,

$$t''^{(m+1)}(f) = t''^{(m)}(f)$$

if $f \notin J^*$.

$$\tau^s = \bigcup \{\tau^{(m)} \mid m < \omega\}.$$

Theorem (Skolem). There exist algorithms ex and un such that

i)
$$\widetilde{\text{ex}}: {}_1F^\tau \to {}_1F^{\tau^s},$$
$$\widetilde{\text{un}}: {}_1F^\tau \to {}_1F^{\tau^s};$$

ii) For any $\varphi \in {}_1F^\tau$, φ is in prenex form,
$\widetilde{\text{ex}}(\varphi)$ does not contain unversal quantifiers,
$\widetilde{\text{un}}(\varphi)$ does not contain existential quantifiers;

iii) For $\varphi \in {}_1F^\tau$ in prenex form

$$M(\varphi) = \mathfrak{S}^\tau \quad \text{iff} \quad M(\widetilde{\text{ex}}(\varphi)) = \mathfrak{S}^{\tau^s},$$

$$M(\varphi) = \emptyset \quad \text{iff} \quad M(\widetilde{\text{un}}(\varphi)) = \emptyset.$$

This theorem is a simple consequence of the Skolem Normal Form Theorem [1, 4, 9].

Let a sentence φ be given in prenex form. Consider the formulae $\widetilde{\text{ex}}(\varphi)$ and $\widetilde{\text{un}}(\varphi)$.

We define the sets P_φ^{ex} (P_φ^{un}) and S_φ^{ex} (S_φ^{un}) as follows:

P_φ^{ex} (P_φ^{un}) is the sequence of zero-order prime formulae of $\widetilde{\text{ex}}(\varphi)$ ($\widetilde{\text{un}}(\varphi)$) which are not sentences i.e., contain at least one free variable. The prime formulae χ and η differing only in free variables are distinguished in P_φ^{ex} (P_φ^{un}).

S_φ^{ex} (S_φ^{un}) is the sequence of zero-order prime sentences of $\widetilde{\text{ex}}(\varphi)$ ($\widetilde{\text{un}}(\varphi)$) if there exist such sentences. In the opposite case, substitute arbitrary constant symbols for the free variables of the elements of P_φ^{ex} (P_φ^{un}) and let the substituted formulae, which are now sentences, be in the sequence S_φ^{ex} (S_φ^{un}).

Example 2. Let the formula φ be

$$(\forall x)(\forall y)(\forall z)[(E(z, x) \leftrightarrow E(z, y)) \leftrightarrow I(x, y)].$$

Then

$$\widetilde{\text{un}}(\varphi) = \varphi,$$

$$\widetilde{\text{ex}}(\varphi) = [E(a, b) \leftrightarrow E(a, c)] \leftrightarrow I(b, c),$$

$$P_\varphi^{\text{un}} = \langle E(z, x), E(z, y), I(x, y)\rangle,$$

$$S_\varphi^{\text{un}} = \langle E(a, a), I(a, a)\rangle$$

for any arbitary constant symbol a,

$$P_\varphi^{\text{ex}} = \emptyset,$$

$$S_\varphi^{\text{ex}} = \langle E(a, b), E(a, c), I(b, c)\rangle.$$

Denote the matrices of $\widetilde{\text{ex}}(\varphi)$ and $\widetilde{\text{un}}(\varphi)$ by $\widetilde{\text{ex}}(\varphi)^*$ and $\widetilde{\text{un}}(\varphi)^*$, respectively.

We recall the concept of unification [8]. Let φ and ψ be prime-formulae. φ and ψ are unifiable iff there exists a substitution such that the substituted φ and ψ are identical literal by literal.

Let φ be a first-order sentence in prenex form. The following generalization of z defines two binary trees $\tilde{z}^{\text{ex}}(\varphi)$ and $\tilde{z}^{\text{un}}(\varphi)$:

Step 1. Start with the formula $\widetilde{\text{ex}}(\varphi)^*$ $(\widetilde{\text{un}}(\varphi)^*)$ to obtain the initial vertex of $\tilde{z}^{\text{ex}}(\varphi)$ (and $\tilde{z}^{\text{un}}(\varphi)$), respectively.

Step 2. Choose the first element of S_ψ^{ex} (S_ψ^{un}), say p, where ψ is the actually-treated vertex and substitute p first by 1 then by 0. Draw two edges from ψ, labelled by p and its negation, respectively.

Seek for the elements of P_ψ^{ex} (P_ψ^{un}) which can be unified with p, and execute the unifying substitution for all such elements.

Step 3. Apply the tautologies listed in § 2 as rules (1)—(18) to obtain the truth-value-free version of the substituted vertex or to obtain a single truth-value along both edges. These truth-value-free formulae or the single truth-values will be the two new vertices χ and η. Then construct the sequences S_χ^{ex} (S_χ^{un}), S_η^{ex} (S_η^{un}), P_χ^{ex} (P_χ^{un}), P_η^{ex} (P_η^{un}).

Step 4. Repeat the steps for every vertices newly obtained if they are not single truth-values. If all of the vertices are truth-values the procedure terminates.

Note that $\tilde{z}^{\text{ex}}(\varphi)$ and $\tilde{z}^{\text{un}}(\varphi)$ exist for any first-order sentence φ if it is in prenex form. The trees do not depend on the order of S_φ^{ex} and S_φ^{un}. If φ is a tautology or it is unsatisfiable then $\tilde{z}^{\text{ex}}(\varphi)$ and $\tilde{z}^{\text{un}}(\varphi)$ are finite trees.

Example 3. Let us construct the trees $\tilde{z}^{\text{ex}}(\varphi)$ and $\tilde{z}^{\text{un}}(\varphi)$ for the formula of Example 2. They are indicated by Fig. 3 and Fig. 4.

Let ${}_1f^{\text{com}}$, ${}_1g^{\text{com}}$ be algorithms such that

i)
$$ {}_1\tilde{f}^{\text{com}}: {}_1F^{\tau^s} \to Z^*, $$
$$ {}_1\tilde{g}^{\text{com}}: {}_1F^{\tau^s} \to Z^*, $$

where elements of ${}_1F^{\tau^s}$ are supposed to be in prenex normal form.

ii)
$$ {}_1\tilde{f}^{\text{com}}(\varphi) = \begin{cases} z_0 & \text{if all the final vertices of } \tilde{z}^{\text{ex}}(\varphi) \text{ are } 1, \\ \text{undefined otherwise,} \end{cases} $$
$$ {}_1\tilde{g}^{\text{com}}(\varphi) = \begin{cases} z_0 & \text{if all the final vertices of } \tilde{z}^{\text{un}}(\varphi) \text{ are } 0. \\ \text{undefined otherwise.} \end{cases} $$

Remark. ${}_1f^{\text{com}}$, ${}_1g^{\text{com}}$ exist by the previous note.

Theorem. Let $\varphi \in {}_1F^{\tau^s}$ be a sentence in prenex form. Then

i) φ is a tautology iff ${}_1\tilde{f}^{\text{com}}(\varphi) = z_0$,

ii) φ is unsatisfiable iff ${}_1\tilde{g}^{\text{com}}(\varphi) = z_0$.

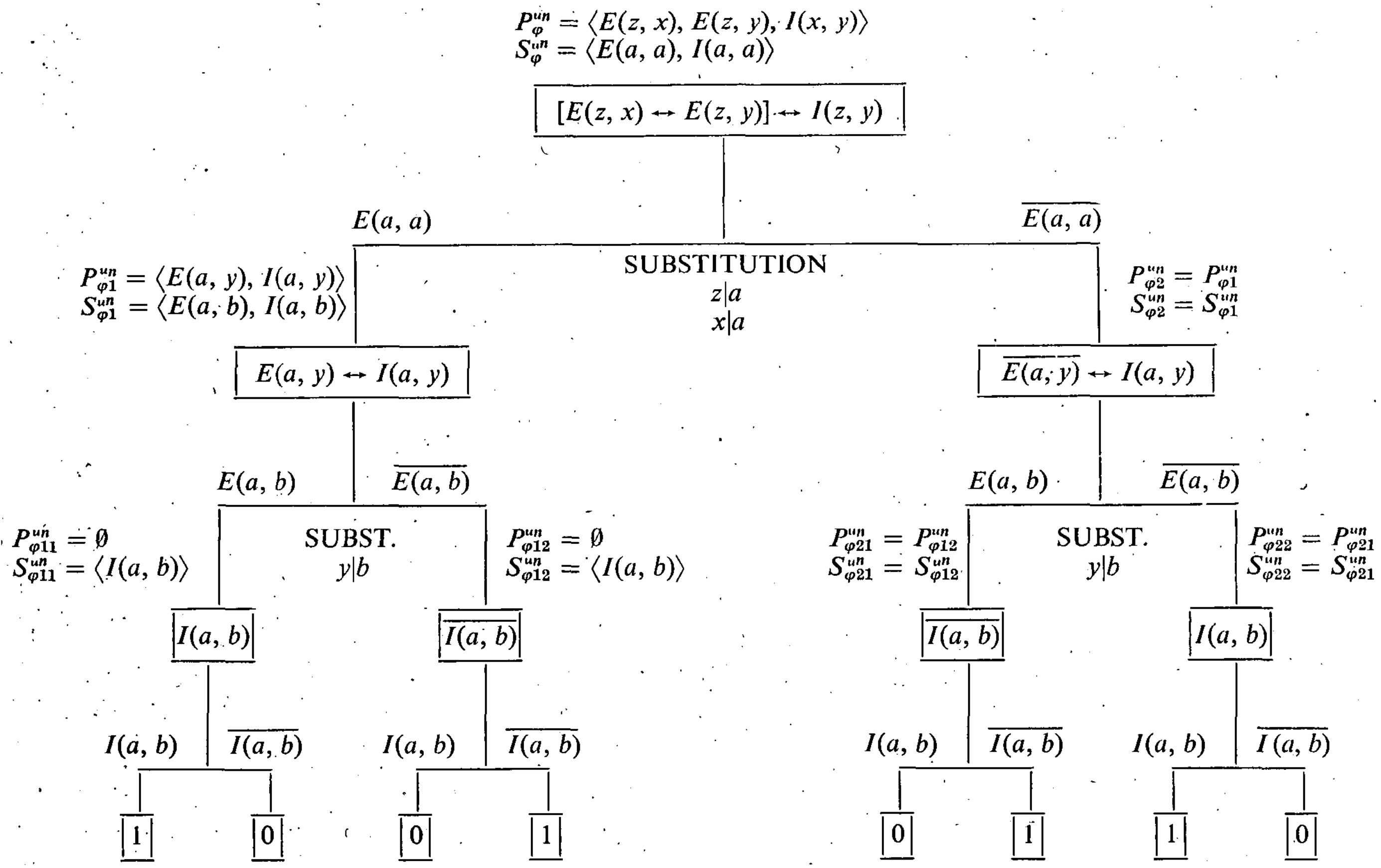
$P^{un}_{\varphi} = \langle E(z, x), E(z, y), I(x, y)\rangle$
$S^{un}_{\varphi} = \langle E(a, a), I(a, a)\rangle$
$[E(z, x) \leftrightarrow E(z, y)] \leftrightarrow I(z, y)$
$E(a, a)$
$\overline{E(a, a)}$
SUBSTITUTION
$z|a$
$x|a$
$P^{un}_{\varphi 1} = \langle E(a, y), I(a, y)\rangle$
$S^{un}_{\varphi 1} = \langle E(a, b), I(a, b)\rangle$
$P^{un}_{\varphi 2} = P^{un}_{\varphi 1}$
$S^{un}_{\varphi 2} = S^{un}_{\varphi 1}$
$E(a, y) \leftrightarrow I(a, y)$
$\overline{E(a, y)} \leftrightarrow I(a, y)$
$E(a, b)$
$\overline{E(a, b)}$
$E(a, b)$
$\overline{E(a, b)}$
SUBST.
$y|b$
SUBST.
$y|b$
$P^{un}_{\varphi 11} = \emptyset$
$S^{un}_{\varphi 11} = \langle I(a, b)\rangle$
$P^{un}_{\varphi 12} = \emptyset$
$S^{un}_{\varphi 12} = \langle I(a, b)\rangle$
$P^{un}_{\varphi 21} = P^{un}_{\varphi 12}$
$S^{un}_{\varphi 21} = S^{un}_{\varphi 12}$
$P^{un}_{\varphi 22} = P^{un}_{\varphi 21}$
$S^{un}_{\varphi 22} = S^{un}_{\varphi 21}$
$I(a, b)$
$\overline{I(a, b)}$
$\overline{I(a, b)}$
$I(a, b)$
$I(a, b)$
$\overline{I(a, b)}$
$I(a, b)$
$\overline{I(a, b)}$
$I(a, b)$
$\overline{I(a, b)}$
$I(a, b)$
$\overline{I(a, b)}$
1
0
0
1
0
1
1
0

Fig. 3
The tree $\tilde{z}^{un}(\varphi)$ of φ

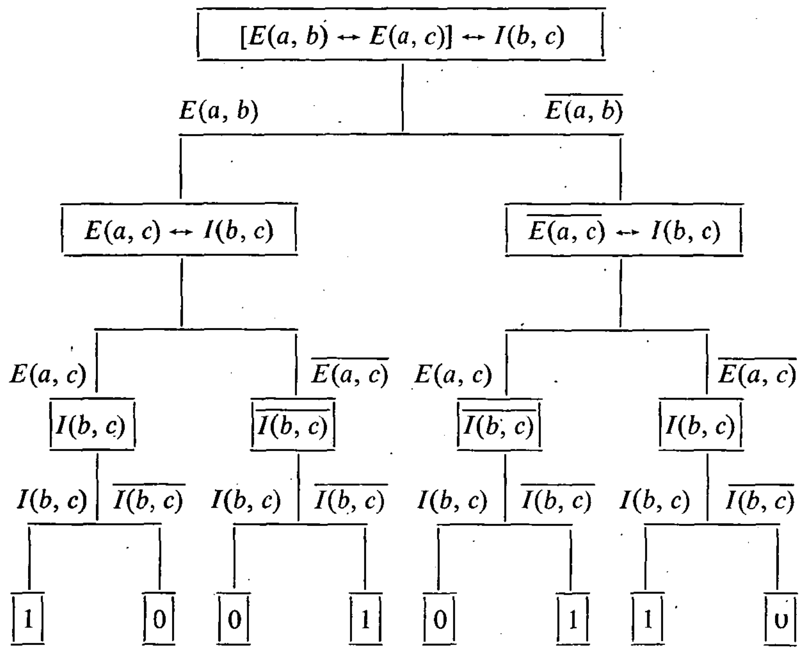

Fig. 4

The tree $\tilde{z}^{ex}(\varphi)$ of φ

Example 4. Let the following sentences be given:

$$\varphi_1 : (\forall x)(A(x) \to (B(x) \wedge C(x))),$$
$$\varphi_2 : (\exists x)(A(x) \wedge D(x)),$$
$$\psi : (\exists x)(D(x) \wedge C(x)).$$

Assume we want to know whether ψ is a consequence of φ_1 and φ_2. There are two possibilities: to show that the formula $\varphi' = \varphi_1 \wedge \varphi_2 \to \psi$ is a tautology; or to show that the formula $\varphi'' = \varphi_1 \wedge \varphi_2 \wedge \neg \psi$ is unsatisfiable. The previous theorem ensures us that if the formula φ' is a tautology then ${}_1\tilde{f}^{com}(\varphi') = z_0$; and if the formula φ'' is unsatisfiable then ${}_1\tilde{g}^{com}(\varphi'') = z_0$.

$$\widetilde{\text{ex}}(\varphi') = (\exists x)(\exists y)(([A(x) \to (B(x) \wedge C(x))] \wedge$$
$$\wedge [A(a) \wedge D(a)]) \to [D(y) \wedge C(y)]),$$

$$\widetilde{\text{un}}(\varphi'') = (\forall x)(\forall y)([A(x) \to (B(x) \wedge C(x))] \wedge$$
$$\wedge [A(a) \wedge D(a)] \wedge [\neg D(y) \vee \neg C(y)]),$$

where a is a Skolem-function (constant).
The tree $\tilde{z}^{ex}(\varphi')$ is indicated in Fig. 5.
All the final-vertices of the tree $\tilde{z}^{ex}(\varphi')$ are 1. Thus ${}_1\tilde{f}^{com}(\varphi') = z_0$, i.e., φ' is a tautology. The tree $\tilde{z}^{un}(\varphi'')$ is very similar to $\tilde{z}^{ex}(\varphi')$ but all the final-vertices are 0. Thus ${}_1\tilde{g}^{com}(\varphi'') = z_0$, i.e. φ'' is unsatisfiable.

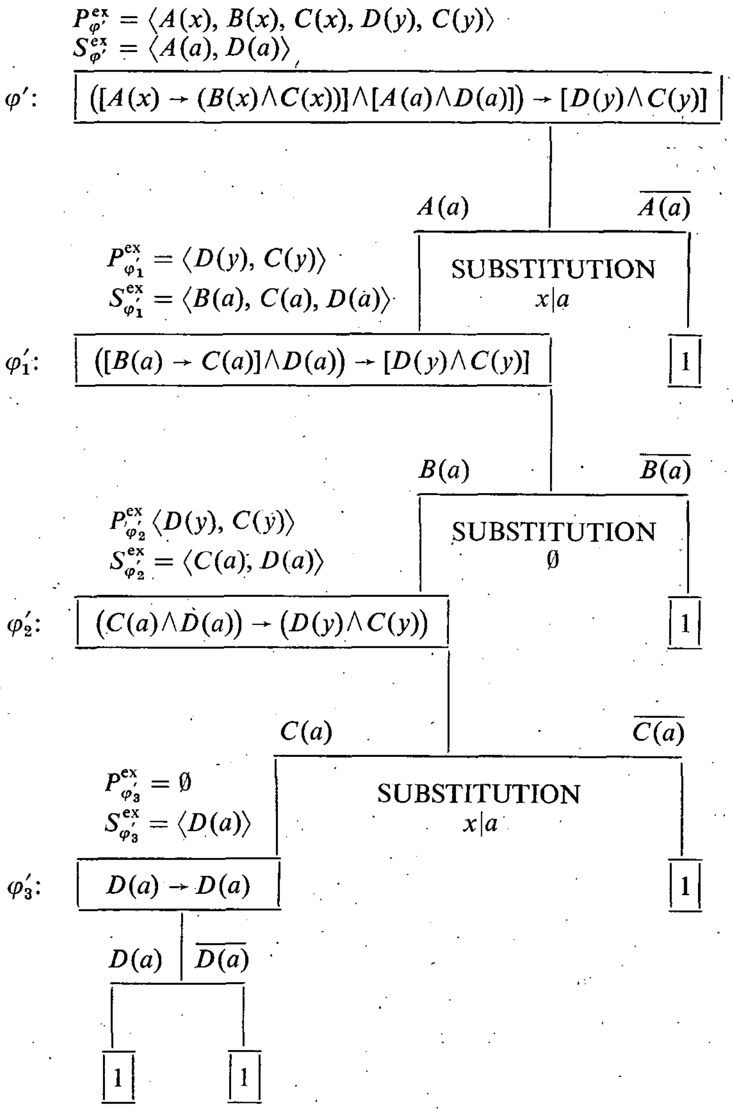
$P^{\text{ex}}_{\varphi'} = \langle A(x), B(x), C(x), D(y), C(y)\rangle$
$S^{\text{ex}}_{\varphi'} = \langle A(a), D(a)\rangle$
φ': $([A(x) \to (B(x) \wedge C(x))] \wedge [A(a) \wedge D(a)]) \to [D(y) \wedge C(y)]$
$A(a)$
$\overline{A(a)}$
$P^{\text{ex}}_{\varphi_1'} = \langle D(y), C(y)\rangle$
$S^{\text{ex}}_{\varphi_1'} = \langle B(a), C(a), D(a)\rangle$
SUBSTITUTION
$x|a$
φ_1': $([B(a) \to C(a)] \wedge D(a)) \to [D(y) \wedge C(y)]$
1
$B(a)$
$\overline{B(a)}$
$P^{\text{ex}}_{\varphi_2'} \langle D(y), C(y)\rangle$
$S^{\text{ex}}_{\varphi_2'} = \langle C(a), D(a)\rangle$
SUBSTITUTION
$\emptyset$
φ_2': $(C(a) \wedge D(a)) \to (D(y) \wedge C(y))$
1
$C(a)$
$\overline{C(a)}$
$P^{\text{ex}}_{\varphi_3'} = \emptyset$
$S^{\text{ex}}_{\varphi_3'} = \langle D(a)\rangle$
SUBSTITUTION
$x|a$
φ_3': $D(a) \to D(a)$
1
$D(a)$
$\overline{D(a)}$
1
1

Fig. 5

The tree $\tilde{z}^{\text{ex}}(\varphi')$

RESEARCH GROUP ON MATHEMATICAL LOGIC AND THEORY OF AUTOMATA OF THE HUNGARIAN ACADEMY OF SCIENCES
H-6720 SZEGED, HUNGARY
SOMOGYI U. 7.

BOLYAI INSTITUTE OF THE ATTILA JÓZSEF UNIVERSITY
H-6720 SZEGED, HUNGARY
ARADI VÉRTANÚK TERE 1.

References

[1] Andréka, H., T. Gergely, I. Németi, Easily comprehensible mathematical logic and its model theory, *KFKI memo*, mimeographed, Budapest, 1975.
[2] Chang, C. L., Theorem proving by generation of pseudo-semantic trees, *Div. of Comput. Res. and Technology*, National Institute of Health, Bethesda, Maryland, 1971.
[3] Chang, C. L., Theorem proving with variable-constrained resolution, *Information Sci.*, v. 4, 1972, pp. 217—231.
[4] Chang, C. C., H. J. Keisler, *Model Theory*, North-Holland Publ. Co., Amsterdam, 1973.
[5] Gödel, K., Die Vollständigkeit der Axiome des logischen Funktionenkalküls, *Monatsh. Math. Phys.*, v. 37, 1930, pp. 349—360.
[6] Herbrand, J., Recherches sur la theorie de la demonstration, *Travaux de la Societe des Sciences et des Lettres de Varsovie*, v. 33, 1930, p. 128.
[7] Prawitz, D., Advances and problems in mechanical proof procedures, *Machine Intelligence*, v. 4, (edited by B. Meltzer and D. Michie), American Elsevier, New York, 1969, pp. 59—71.
[8] Robinson, J. A., A machine oriented logic based on the resolution principle, *J. Assoc. Comput. Mach.*, v. 1965, pp. 23—41.
[9] Skolem, T., Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit oder Beweisbarkeit mathematischer Sätze nebst einem Theorem über dichte Mengen, *Skrifter utgitt av Videnskapsselskapet i Kristiania, I, Mat. Naturv. Kl.*, v. 4, 1920, p. 36.
[10] Turing, A. M., Computing machinery and intelligence, *Mind*, v. 59, 1950, pp. 433—460.
[11] Varga, A., P. Ecsedi-Tóth, F. Móricz, An effective method for minimizing Boolean polynomials, *Research Report*, Research Group on Mathematical Logic and Theory of Automata of the Hungarian Academy of Sciences, 1977, p. 40.

(Received May 17, 1977)