# Checking the Four-Valued Boolean Algebra by the Use PLANNER

Vitaly Eltekov

Moscow M.V.Lomonosov State University, Physics Department
119234 Moscow, Russia
eltekov@simula.phys.msu.su

**Abstract.** A package for testing the Boolean algebra relationships is proposed. The language of implementation is Planner. The same language is used as input one. The output digital data are presented in form of lists, and the Boolean relations issued in LaTeX notation. The consideration is based on four-valued Boolean algebra. It connects to an attempt to resolve Russell's paradox taking into consideration such properties of logical assertion as properties to be 'non-truth' and 'non-false' or 'truth' and 'false' simultaneously, besides usual properties to be 'truth' or 'false'. In thus obtained logic the number of possibilities to connect couples of such variables is close to 4.5 billions instead of 16 in usual two-valued logic. Some matrix representations of these connections and algebraic relationships are introduced. Formerly the relationships, which took place in single-valued and double-valued logics, were affirmed. Then the formulas of expanded four-valued logic were considered and set of new tautologies were confirmed. This logic allows to extend some relationships by introducing the alternative functions of logic, which do not obey the Frege axioms, but are relative to corresponding functions of expanded logic. In particular the reflexive law of implication and equivalence is invalid in such an extended logic. New form of the law of excluded middle appears in this logic. Such a form gives a possible explanation of Russell's paradox. Many tautologies, which take place in expanded logic, transform to extended ones. ...

## 1 Introduction

An attempt to resolve the Russell's paradox [1] (see also [2]) may be connected with taking into consideration such properties of logical assertion as property to be either 'true' and 'false' ones simultaneously, or 'non-true' and 'non-false' simultaneously as well. In the case of Boolean algebra these properties are vtrue' and 'false'. This leads to conclusion that propositional variables (see [3]) may have 4 values. In such a way obtained algebra of logic the number of binary relations between the pair of variables is large enough, $4^{4^2} = 4,294,967,296$ instead of $2^{2^2} = 16$ in usual two-valued logic. Let us consider the main formulas of mono-valued (positive) and two-valued logics.

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2657, pp. 764–773, 2003.

## 2 Positive Logic

Let us define the positive logic as a system of rules connecting assertions which have only property 'true'. Among these rules one can extract the modus ponens and one of syllogism modi named 'barbara' (see [4], for example). Let $A, B, C$ be some assertions. Modus ponens says that if $A$ takes place and $A$ implies $B$, then $B$ takes place as well. Modus 'barbara' says that if $A$ implies $B$ and $B$ implies $C$, then $A$ implies $C$.

The PLANNER language [5] allows to solve some problems of logical deduction, which are based, in particular, on usage of these modi. The basic part of this language is a development of well-known language LISP (see [6], for example). Though PLANNER is a functional language, it allows to use such traditional means for algorithm composing as programming with jumps, cycles, and compound operators. In addition to these opportunities there is a way for solving the problems of logical inference defining the special functions, theorems, which contain in their head a pattern, besides the arguments. This pattern serves as a label for theorem call, backtracking and pattern matching for data base [5] being applied here as a rule.

Let us consider the next example. Lets data base contain the following assertions: (A1 implies A2), (A2 implies A3), ... (AN-1 implies AN). It is necessary to prove that due to syllogism modus the consequence (A1 implies AN) takes place. For this aim let us define the theorem:

[DEFINE chain (CONSEQ (x,y,z) (*x implies *z)

[SEARCH (.x implies *y)] [GOAL (.y implies .z)] )]

Here the words DEFINE and CONSEQ show that a theorem is defined. Its pattern (*x implies *z) contains re-defined variables x and y, prefix * indicating on it. The word 'implies' is fixed one. Function SEARCH tries to find assertion (.x implies *y) in data base with fixed value of x and undefined value of y. Next function GOAL, calling just defined theorem, finds the similar assertion, but with fixed value of y. Suppose N to be 750, for example. Then the theorem is proved using the function call

[GOAL (A1 implies A750)].

The proof requires less than 10 seconds if the computer has performance as 400 MHz. Checking for another values of $N$ showed that elapsed time is proportional to $N^2$.

The PLANNER-B package, which was applied to checking the formulas many-valued logics (see below), is analogous to PLANNER-ANALYTIC package [7]. Its kernel is a set of functions for recursive pattern matching [8], which allows to organize sufficiently compact algorithm recording.

## 3 Two-Valued Logic

If considering assertions might be false, then they would be manipulated according to the rules of Boolean algebra (see [3], p.e.). Objects of this algebra are

the variables, which may take only one of two values, 'true' or 'false'. The functions of such arguments take one of these values as well. The functions having value 'true' for any values of arguments are named as tautologies. Hilbert and Bernays [9] successively defined the relationships between the objects of Boolean algebra on the base of set of axioms. For these axioms were choken by use the package PLANNER-B we list them in full.

I. Formulas for implication:

$$A \to (B \to A)\,, \tag{1}$$

$$(A \to (A \to B)) \to (A \to B)\,, \tag{2}$$

$$(A \to B) \to ((B \to C) \to (A \to C))\,. \tag{3}$$

II. Formulas for conjunction:

$$(A \wedge B) \to A\,, \tag{4}$$

$$(A \wedge B) \to B\,, \tag{5}$$

$$(A \to B) \to ((A \to C) \to (A \to (B \wedge C)))\,. \tag{6}$$

III. Formulas for disjunction:

$$A \to (A \vee B)\,, \tag{7}$$

$$B \to (A \vee B)\,, \tag{8}$$

$$(A \to C) \to ((B \to C) \to ((A \vee B) \to C))\,. \tag{9}$$

IV. Formulas for equivalence:

$$(A \equiv B) \to (A \to B)\,, \tag{10}$$

$$(A \equiv B) \to (B \to A)\,, \tag{11}$$

$$(A \to B) \to ((B \to A) \to (A \equiv B))\,. \tag{12}$$

V. Formulas for negation:

$$(A \to B) \to (\neg B \to \neg A)\,, \tag{13}$$

$$A \to \neg\neg A\,, \tag{14}$$

$$\neg\neg A \to A\,. \tag{15}$$

All these formulas must be tautologies.

As an example let us try to prove the reflexive law (Aristotle law), $A \to A$, using Frege axioms

$$P_1 \to (P_2 \to P_1), \tag{16}$$

$$(P_1 \to (P_2 \to P_3)) \to ((P_1 \to P_2) \to (P_1 \to P_3)) \tag{17}$$

as source for proof.

Let functions FA1 and FA2 write these formulas to data base:
[DEFINE FA1 (LAMBDA (a b) [ASSERT (⊢ (.a → (.b → .a)))] )]
[DEFINE FA2 (LAMBDA (a b c)
[ASSERT (⊢ ((.a → (.b → .c)) → ((.a → .b) → (.a → .c))))] )]
Lets the theorem, defined below, allow achieve the goal corresponding to its pattern:
[DEFINE Modus-Ponens (CONSEQ (a b) (⊢ *b)
[SEARCH (⊢ (*a → .b))] [SEARCH (⊢ .a)] [ASSERT (⊢ .b)] )].
To fill data base it is necessary to call the program, which makes all possible substitutions A and (A → A) instead of formal parameters:
[PROG (k1 k2 k3 k4 k5 (l (A (A → A))))
[LOOP k1 .l [LOOP k2 .l [FA1 .k1 .k2] ]]
[LOOP k3 .l [LOOP k4 .l [LOOP k5 .l [FA2 .k3 .k4 .k5] ]]] ]
Before invoking the Modus-Ponens theorem we should define and apply the following theorem,
[DEFINE M-P-2 (CONSEQ (a b c) (modus ponens *c)
[SEARCH (⊢ (*a → *b))] [SEARCH (⊢ .a)]
[ASSERT (⊢ .b)] [ACHIEVE (⊢ .c)] )]
[ACHIEVE (modus ponens (A → A))],
which creates additive assertions, in particular,

$$\vdash ((A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))). \tag{18}$$

This relation is a key to solution.
The solution is obtained when the former theorem is invoked,
[GOAL (⊢ (A → A)) ].
In the case of two-valued logic the formulas I-V allow to obtain the value matrices for corresponding functions. These matrices may also be obtained by another way.
Let Boolean 'true' and 'false' correspond to numbers 1 and -1. Let $x$ and $y$ take values either 1, or -1. The functions of $x$ and $y$, which take values either 1, or -1, are introduced:

$$\beta_{\to}(x, y) = (xy - x + y + 1)/2 \,, \tag{19}$$

$$\beta_{\leftarrow}(x, y) = (xy + x - y + 1)/2 \,, \tag{20}$$

$$\beta_{\vee}(x, y) = (-xy + x + y + 1)/2 \,, \tag{21}$$

$$\beta_{\wedge}(x, y) = (xy + x + y - 1)/2 \,, \tag{22}$$

$$\beta_{\equiv}(x, y) = xy \,, \tag{23}$$

$$\beta_{\neg}(x) = -x \,. \tag{24}$$

These functions correspond to the following logical relations: implication, inverse implication, disjunction, conjunction, equivalence, and negation.
All regulations, which take place in the positive logic, are written as formulas of the two-valued logic. In particular the next formulas will be tautologies:

Modus ponens:

$$(A \wedge (A \to B)) \to B \,. \tag{25}$$

Syllogism modus:

$$((A \to B) \wedge (B \to C)) \to (A \to C) \,. \tag{26}$$

In contrast to the case of the positive logic here the variables may have 'true' or 'false' values. Thus the truth of formula does not prove the truth of variables participating in it.

The truth of the following formalas,

$$(A \to (A \to B)) \to (A \to B) \,, \tag{27}$$

$$(A \to (B \to C)) \to (B \to (A \to C)) \,, \tag{28}$$

$$(A \to (B \to C)) \to ((A \to B) \to (A \to C)) \,, \tag{29}$$

$$(A \to B) \to ((A \to (B \to C)) \to (A \to C)) \,, \tag{30}$$

is hardly manifested by means of the positive logic possibilities, but in the frames of two-valued logic it is simply tested by use the substitution.

The next laws take place:

The law of excluded third,

$$\neg(A \wedge \neg A) \,, \tag{31}$$

the reflexive laws,

$$A \to A \quad \text{and} \quad A \equiv A \,, \tag{32}$$

the distributive law for implication,

$$(A \to (B \to C)) \to ((A \to B) \to (A \to C)) \,, \tag{33}$$

The 'reductio ad absurdum' law,

$$(\neg A \to B) \to ((\neg A \to \neg B) \to A) \,, \tag{34}$$

De Morgan laws,

$$(\neg(A \wedge B)) \equiv ((\neg A) \vee (\neg B)) \,, \tag{35}$$

$$(\neg(A \vee B)) \equiv ((\neg A) \wedge (\neg B)) \,, \tag{36}$$

the distributive laws,

$$((A \vee B) \wedge C) \equiv ((A \wedge C) \vee (B \wedge C)) \,, \tag{37}$$

$$((A \wedge B) \vee C) \equiv ((A \vee C) \wedge (B \vee C)) \,. \tag{38}$$

All these relationships are well-known (see [10]). In the next section the tests of above formulas as well as additive formulas in the frame of four-valued logic are discribed.

## 4 Four-Valued Expanded Logic

The formulas for functions $\beta(x, y)$ conserve their applicability, if the 'real' unity $e$ ($e^2 = 1$), which is not a number, is introduced. Then $1, e, -e, -1$ may be the propositional values for variables and functions. These values may be related to the numbers 1,2,3,4 on the other hand. The next matrices will be correspond to the functions introduced above:

| $\beta_{\rightarrow}(x,y)$ | $\beta_{\leftarrow}(x,y)$ | $\beta_{\vee}(x,y)$ | $\beta_{\wedge}(x,y)$ | $\beta_{\equiv}(x,y)$ |
|---|---|---|---|---|
| 1 2 3 4 | 1 1 1 1 | 1 1 1 1 | 1 2 3 4 | 1 2 3 4 |
| 1 1 3 3 | 2 1 2 1 | 1 2 1 2 | 2 2 4 4 | 2 1 4 3 |
| 1 2 1 2 | 3 3 1 1 | 1 1 3 3 | 3 4 3 4 | 3 4 1 2 |
| 1 1 1 1 | 4 3 2 1 | 1 2 3 4 | 4 4 4 4 | 4 3 2 1 |

In the case of four-valued logic the matrices for functions $\beta(x, y)$ may be obtained using the tautologies I-V of precedent section. In contrast to the case of two-valued logic the tautologies of the group I give as a result 6351 variants for values of matrix $\beta_{\rightarrow}(x, y)$. If one chooses the variant, which is defined by the table above, then another matrices will be simply obtained according to formulas II-IV.

In this four-valued interpretation of Boolean algebra three negations appeared: $\beta_{\neg}(x) = -x \sim (4, 3, 2, 1)$, $\beta_{\top}(x) = ex \sim (2, 1, 4, 3)$, $\beta_{\bot}(x) = -ex \sim (3, 4, 1, 2)$.

The following relationships take place:

$$\neg\neg A \equiv \top\top A \equiv \bot\bot A \equiv A\ , \tag{39}$$

$$\neg\top A \equiv \bot A, \top\bot A \equiv \neg A, \bot\neg A \equiv \top A\ . \tag{40}$$

The law of excluded middle has now two forms:

$$\neg(A \wedge \neg A) \quad \text{and} \quad \neg(\top A \wedge \bot A)\ . \tag{41}$$

The following definitions are introduced:

$$(A \leftarrow B) \equiv ((\neg A) \rightarrow (\neg B))\ , \tag{42}$$

$$(A \uparrow B) \equiv ((\top A) \rightarrow (\top B))\ , \tag{43}$$

$$(A \downarrow B) \equiv ((\bot A) \rightarrow (\bot B))\ . \tag{44}$$

Then 'modus ponens' formula is generalized as following:

$$(A \wedge (A \rightarrow B)) \rightarrow B, \quad (\neg A \wedge (A \leftarrow B)) \rightarrow \neg B\ , \tag{45}$$

$$(\top A \wedge (A \uparrow B)) \rightarrow \top B, \quad (\bot A \wedge (A \downarrow B)) \rightarrow \bot B\ . \tag{46}$$

Syllogism modus takes the following four forms:

$$((A \to B) \wedge (B \to C)) \to (A \to C)\,, \tag{47}$$

$$((A \leftarrow B) \wedge (B \leftarrow C)) \to (A \leftarrow C)\,, \tag{48}$$

$$((A \uparrow B) \wedge (B \uparrow C)) \to (A \uparrow C)\,, \tag{49}$$

$$((A \downarrow B) \wedge (B \downarrow C)) \to (A \downarrow C)\,. \tag{50}$$

Also the reflexive law has four types:

$$A \to A, \quad A \leftarrow A, \quad A \uparrow A, \quad A \downarrow A\,. \tag{51}$$

Next formulas will be tautologies:

$$((P \to Q) \wedge (Q \to P)) \equiv (P \equiv Q), \quad ((P \leftarrow Q) \wedge (Q \leftarrow P)) \equiv (P \equiv Q)\,, \tag{52}$$

$$((P \uparrow Q) \wedge (Q \uparrow P)) \equiv (P \equiv Q), \quad ((P \downarrow Q) \wedge (Q \downarrow P)) \equiv (P \equiv Q)\,. \tag{53}$$

## 5 Four-Valued Extended Logic

The next type of connection of propositional variables values is based on a modification of mentioned above algebraic formulas. The function $\epsilon(x, y)$ substitutes unity in formulas for $\beta$ :

$$\epsilon(x, y) = (1 + x^2 + y^2 - x^2y^2)/2\,. \tag{54}$$

Then the numbers $1, i, -i, -1$ $(i^2 = -1)$ are taken as admitted values. They may be allowed to correspond to numbers 1,2,3,4. The following matrices will correspond to such introduced functions:

| $\beta_{\tilde{\to}}(x, y)$ | $\beta_{\tilde{\leftarrow}}(x, y)$ | $\beta_{\tilde{\vee}}(x, y)$ | $\beta_{\tilde{\wedge}}(x, y)$ | $\beta_{\tilde{\equiv}}(x, y)$ |
|---|---|---|---|---|
| 1 2 3 4 | 1 1 1 1 | 1 1 1 1 | 1 2 3 4 | 1 2 3 4 |
| 1 4 3 3 | 2 4 2 1 | 1 2 4 2 | 2 2 1 4 | 2 4 1 3 |
| 1 2 4 2 | 3 3 4 1 | 1 4 3 3 | 3 1 3 4 | 3 1 4 2 |
| 1 1 1 1 | 4 3 2 1 | 1 2 3 4 | 4 4 4 4 | 4 3 2 1 |

In this alternative interpretation of Boolean algebra three negations do appear: $\beta_{\neg}(x) = -x \sim (4, 3, 2, 1)$, $\beta_{\tilde{\top}}(x) = ix \sim (2, 4, 1, 3)$,
$\beta_{\tilde{\bot}}(x) = -ix \sim (3, 1, 4, 2)$.

In this interpretation the law of excluded middle takes place (but not excluded third):

$$\neg(A\tilde{\wedge}\neg A\tilde{\wedge}\tilde{\top}A\tilde{\wedge}\tilde{\bot}A)\,. \tag{55}$$

The designations are introduced:

$$(P\ \tilde{\uparrow}Q) \equiv (\tilde{\top}P\tilde{\to}\tilde{\top}Q), \quad (P\ \tilde{\downarrow}Q) \equiv (\tilde{\bot}P\tilde{\to}\tilde{\bot}Q)\,. \tag{56}$$

The reflexive law transforms as followed:

$$(A\tilde{\rightarrow}A)\tilde{\vee}(A\tilde{\uparrow}A),\quad (A\tilde{\rightarrow}A)\tilde{\vee}(A\tilde{\downarrow}A),\quad (A\tilde{\leftarrow}A)\tilde{\vee}(A\tilde{\uparrow}A),\quad (A\tilde{\leftarrow}A)\tilde{\vee}(A\tilde{\downarrow}A)\ . \tag{57}$$

Modus ponens is generalized by the following way:

$$((A\tilde{\wedge}(A\tilde{\rightarrow}B))\tilde{\rightarrow}B)\tilde{\vee}((\tilde{\top}A\tilde{\wedge}(A\tilde{\uparrow}B))\tilde{\rightarrow}\tilde{\top}B)\ , \tag{58}$$

$$((\neg A\tilde{\wedge}(A\tilde{\leftarrow}B))\tilde{\rightarrow}\neg B)\tilde{\vee}((\tilde{\top}A\tilde{\wedge}(A\tilde{\uparrow}B))\tilde{\rightarrow}\tilde{\top}B)\ , \tag{59}$$

$$((A\tilde{\wedge}(A\tilde{\rightarrow}B))\tilde{\rightarrow}B)\tilde{\vee}((\tilde{\bot}A\tilde{\wedge}(A\tilde{\downarrow}B))\tilde{\rightarrow}\tilde{\bot}B)\ , \tag{60}$$

$$((\neg A\tilde{\wedge}(A\tilde{\leftarrow}B))\tilde{\rightarrow}\neg B)\tilde{\vee}((\tilde{\bot}A\tilde{\wedge}(A\tilde{\downarrow}B))\tilde{\rightarrow}\tilde{\bot}B)\ . \tag{61}$$

Syllogism modus takes four forms:

$$(((A\tilde{\rightarrow}B)\tilde{\wedge}(B\tilde{\rightarrow}C))\rightarrow(A\tilde{\rightarrow}C))\tilde{\vee}(((A\tilde{\uparrow}B)\tilde{\wedge}(B\tilde{\uparrow}C))\rightarrow(A\tilde{\uparrow}C))\ , \tag{62}$$

$$(((A\tilde{\leftarrow}B)\tilde{\wedge}(B\tilde{\leftarrow}C))\rightarrow(A\tilde{\leftarrow}C))\tilde{\vee}(((A\tilde{\uparrow}B)\tilde{\wedge}(B\tilde{\uparrow}C))\rightarrow(A\tilde{\uparrow}C))\ , \tag{63}$$

$$(((A\tilde{\rightarrow}B)\tilde{\wedge}(B\tilde{\rightarrow}C))\rightarrow(A\tilde{\rightarrow}C))\tilde{\vee}(((A\tilde{\downarrow}B)\tilde{\wedge}(B\tilde{\downarrow}C))\rightarrow(A\tilde{\downarrow}C))\ , \tag{64}$$

$$(((A\tilde{\leftarrow}B)\tilde{\wedge}(B\tilde{\leftarrow}C))\rightarrow(A\tilde{\leftarrow}C))\tilde{\vee}(((A\tilde{\downarrow}B)\tilde{\wedge}(B\tilde{\downarrow}C))\rightarrow(A\tilde{\downarrow}C))\ . \tag{65}$$

The reflexive law has four forms as well:

$$(A\tilde{\rightarrow}A)\tilde{\vee}(A\tilde{\uparrow}A),\quad (A\tilde{\leftarrow}A)\tilde{\vee}(A\tilde{\uparrow}A),\quad (A\tilde{\rightarrow}A)\tilde{\vee}(A\tilde{\downarrow}A),\quad (A\tilde{\leftarrow}A)\tilde{\vee}(A\tilde{\downarrow}A)\ . \tag{66}$$

Next formulas will be tautologies:

$$((P\tilde{\rightarrow}Q)\tilde{\wedge}(Q\tilde{\rightarrow}P))\equiv(P\tilde{\equiv}Q),\quad ((P\tilde{\leftarrow}Q)\tilde{\wedge}(Q\tilde{\leftarrow}P))\equiv(P\tilde{\equiv}Q)\ , \tag{67}$$

$$((P\tilde{\uparrow}Q)\tilde{\wedge}(Q\tilde{\uparrow}P))\equiv\neg(P\tilde{\equiv}Q),\quad ((P\tilde{\downarrow}Q)\tilde{\wedge}(Q\tilde{\downarrow}P))\equiv\neg(P\tilde{\equiv}Q)\ . \tag{68}$$

The formulas, which are analogous to formulas for implication (sec. 2), formulas 'reductio ad absurdum' are not tautologies in this system of relationships. Nevertheless they may be represented in extended form (see sec.7).

## 6 An Example: Proof of Possibility of Russell's Set Existence

One of the aims of this work was attempt to resolve the paradox connected with Russell's set. The Russell's set $R$ [1] is defined by formula:

$$R\ =\ \{x|x\notin x\}. \tag{69}$$

Taking the axiom

$$z\in\{y|\phi(y)\}\equiv\phi(z) \tag{70}$$

into account, we have

$$u\in R\ \equiv\ u\notin u. \tag{71}$$

Substitution $R$ instead of $u$ gives

$$R \in R \ \equiv \ R \notin R. \tag{72}$$

On the other hand, the reflexive law says:

$$R \in R \ \equiv \ R \in R. \tag{73}$$

In the frame of two-valued logic the supposion

$$\neg(x \in x) \ \equiv \ x \notin x \tag{74}$$

takes place. This gives

$$R \in R \ \equiv \ \neg(R \notin R), \tag{75}$$

and then leads to

$$\neg(R \in R \ \equiv \ R \notin R). \tag{76}$$

The relationships (75) and (76) contradict one another.

In the frame of four-valued logic the situation is changed extremely. We suggest that relations $R \in R$ and $R \notin R$ are objects of extended four-valued logic. The equivalence, as logical relation, should be generalized by the following way:

$$(A \tilde{\equiv} B) \tilde{\vee} ((\tilde{\top} A) \tilde{\equiv} (\tilde{\top} B)) \tag{77}$$

This formula will be a tautology when $A \sim B$ and $A \sim \neg B$. Thus formulas

$$(A \tilde{\equiv} A) \tilde{\vee} ((\tilde{\top} A) \tilde{\equiv} (\tilde{\top} A)) \tag{78}$$

and

$$(A \tilde{\equiv} (\neg A)) \tilde{\vee} ((\tilde{\top} A) \tilde{\equiv} (\tilde{\top} (\neg A))) \tag{79}$$

are tautologies. Substituting $R \in R$ instead of $A$ we resolve the contradiction above.

## 7 Some Analogies in Considered Types of Logic

Confirmation of implication (Frege axiom):

$$\begin{gathered}
A \rightarrow \vdash (B \rightarrow A), \ A \uparrow \top (B \uparrow A), \\
A \rightarrow \neg (B \leftarrow A), \ A \downarrow \bot (B \downarrow A), \\
(A \tilde{\rightarrow} \vdash (B \tilde{\rightarrow} A)) \ \tilde{\vee} \ (A \tilde{\uparrow} \tilde{\top} (B \tilde{\uparrow} A)), \\
(A \tilde{\rightarrow} \vdash (B \tilde{\rightarrow} A)) \ \tilde{\vee} \ (A \tilde{\downarrow} \tilde{\bot} (B \tilde{\downarrow} A)), \\
(A \tilde{\leftarrow} \neg (B \tilde{\leftarrow} A)) \ \tilde{\vee} \ (A \tilde{\uparrow} \tilde{\top} (B \tilde{\uparrow} A)), \\
(A \tilde{\leftarrow} \neg (B \tilde{\leftarrow} A)) \ \tilde{\vee} \ (A \tilde{\downarrow} \tilde{\bot} (B \tilde{\downarrow} A)).
\end{gathered} \tag{80}$$

The modified 'reductio ad absurdum' law:

$$\begin{gathered}((A \to B) \wedge (A \to \neg B)) \equiv \neg A,\ ((A \uparrow B) \wedge (A \uparrow \neg B)) \equiv \bot A,\\ ((A \leftarrow B) \wedge (A \leftarrow \neg B)) \equiv \vdash A,\ ((A \downarrow B) \wedge (A \downarrow \neg B)) \equiv \top A,\\ (((A \tilde{\to} B) \tilde{\wedge} (A \tilde{\to} \neg B)) \equiv \neg A) \ \tilde{\vee}\ (((A \tilde{\uparrow} B) \tilde{\wedge} (A \tilde{\uparrow} \neg B)) \equiv \tilde{\bot} A),\\ (((A \tilde{\to} B) \tilde{\wedge} (A \tilde{\to} \neg B)) \equiv \neg A) \ \tilde{\vee}\ (((A \tilde{\downarrow} B) \tilde{\wedge} (A \tilde{\downarrow} \neg B)) \equiv \tilde{\top} A),\\ (((A \tilde{\leftarrow} B) \tilde{\wedge} (A \tilde{\leftarrow} \neg B)) \equiv \vdash A) \ \tilde{\vee}\ (((A \tilde{\uparrow} B) \tilde{\wedge} (A \tilde{\uparrow} \neg B)) \equiv \tilde{\bot} A),\\ (((A \tilde{\leftarrow} B) \tilde{\wedge} (A \tilde{\leftarrow} \neg B)) \equiv \vdash A) \ \tilde{\vee}\ (((A \tilde{\downarrow} B) \tilde{\wedge} (A \tilde{\downarrow} \neg B)) \equiv \tilde{\top} A). \qquad (81)\end{gathered}$$

## 8 Conclusion

The approbation of some formulas of four-valued logic was performed. Few types of formulas of expanded and extended logics were designated. In the expanded logic all relationships, which took place in two-valued algebra, were affirmed. The additive relationships, which appeared in this case, were choken as well. In the extended logic only part of tautologies related to the expanded one took place. In particular the reflexive law of implication and equivalence was invalid. In the case of extended logic some tautologies, which were analogous to corresponding tautologies of expanded logic, were composed. These tautologies allow to develop the additive rules of logical deduction besides conventional ones. The approbation performed could be taken into account while developing the principles of automated proving (see [11]).

## References

1. Russell B.: The Principles of Mathematics (1903)
2. Lavrov, S.S.: On a relation between the bases of programming and mathematics. Programming N 6 (2001) 3–12
3. Mendelson, E.: Introduction to Mathematical Logic. Princeton Toronto New York London (1964)
4. Kolmogorov, A.N., Dragalin, A.G.: Introduction to Mathematical Logic. Moscow Univ. Publ. (1982)
5. Pilschikov, V.N.: PLANNER language. Nauka, Moscow (1983)
6. Lavrov, S.S., Silagadze, G.S.: Automated data processing. LISP language and its realization. Nauka, Moscow (1978)
7. Eltekov, V.A.: Program package PLANNER-ANALYTIC. In: Samarskii, A.A. (ed): Applied Program Packages. Analytic Transformations. Nauka, Moscow (1988) 147–151
8. Shaposhnikov, N.N., Eltekov, V.A.: On a method for computer realization of analytic transformations. Programmirovanie, N 5 (1986) 84–87
9. Hilbert, D., Bernays, P.: Grundlagen der Mathematik, Bd. I. Berlin (1934)
10. Church, A.: Introduction to Mathematical Logic. Vol. I. Princeton University Press (1956)
11. Lauriere, J.-L.: Intelligence Artificielle. Eyrolles, Paris (1987)