

**M.Lvov**

**About one Approach for Designing of Algebraic Computations:  
Computations in Boolean Algebra.**

Designing of algorithms for **algebraic computations** – the main task arising when realising mathematical systems based on symbolic transformations

**Mathematical model** for this problem – **multisorted algebraic systems (MAS)**

**The process of realising of algebraic computations needs carefull preliminary design MAS as hierarchy of its sorts and specifications of algebraic operations interpreters**

## Theory and details in:

- **Lvov M.S.** Synthesis of Interpreters of Algebraic Operations in Extensions of Multisorted Algebras. (ukr, prepared for print)
- **Lvov M.S.** Verification of Interpreters of Algebraic Operations in Extensions of Multisorted Algebras. (prepared for print)
- **Lvov M.S.** Method of Inheritance for design of algebraic computations in mathematical educational systems. (ukr, prepared for print)
- **Lvov M.S.** Method of Morphisms for design of algebraic computations in mathematical educational systems. (ukr, prepared for print)

## Approach **IEM**

**I**nheritance, **E**xtensions, **M**orphisms

Base principles and ideas of IEM are quite simple and well known in mathematics and programming

Boolean algebra is very **simple, well known and interesting** example of **subject domain** for applying of **IEM**

## **Algebraic programming system APS**

**(V. Peschanenko)**

**APS uses technologies of algebraic programming, based on rewriting rules systems and strategies of rewriting.**

**The interpreter of algebraic operation defined by rewriting rules system.**

**We shall consider problems:**

- of design,**
- synthesis,**
- verification**

**of interpreters of boolean algebra operations (logical operations).**

## Problem Definition

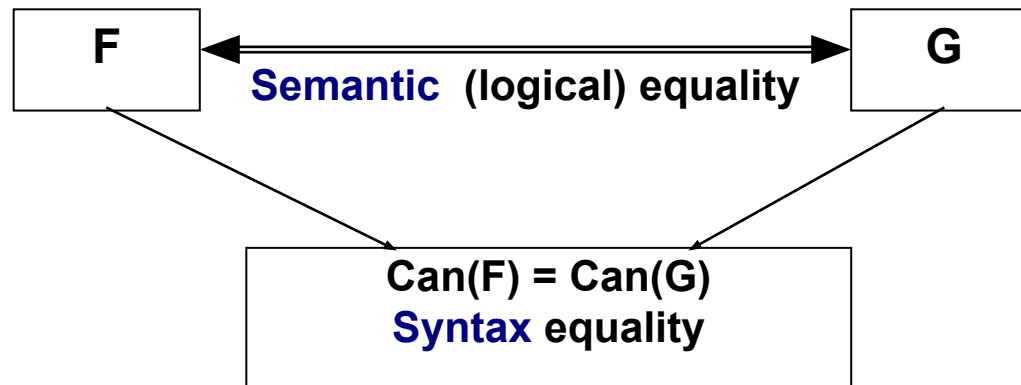
Compute the value  $Val(F(x_1, \dots, x_n))$  of logical expression  $F(x_1, \dots, x_n)$   
in signature of logical operations  $\langle \&, \vee, \neg, 0, 1 \rangle$

By the value of expression  $F(x_1, \dots, x_n)$  we means its canonical form  
i.e. such expression  $Can(F)(x_1, \dots, x_n)$  that

$$F(x_1, \dots, x_n) \equiv Can(F)(x_1, \dots, x_n) \quad (1)$$

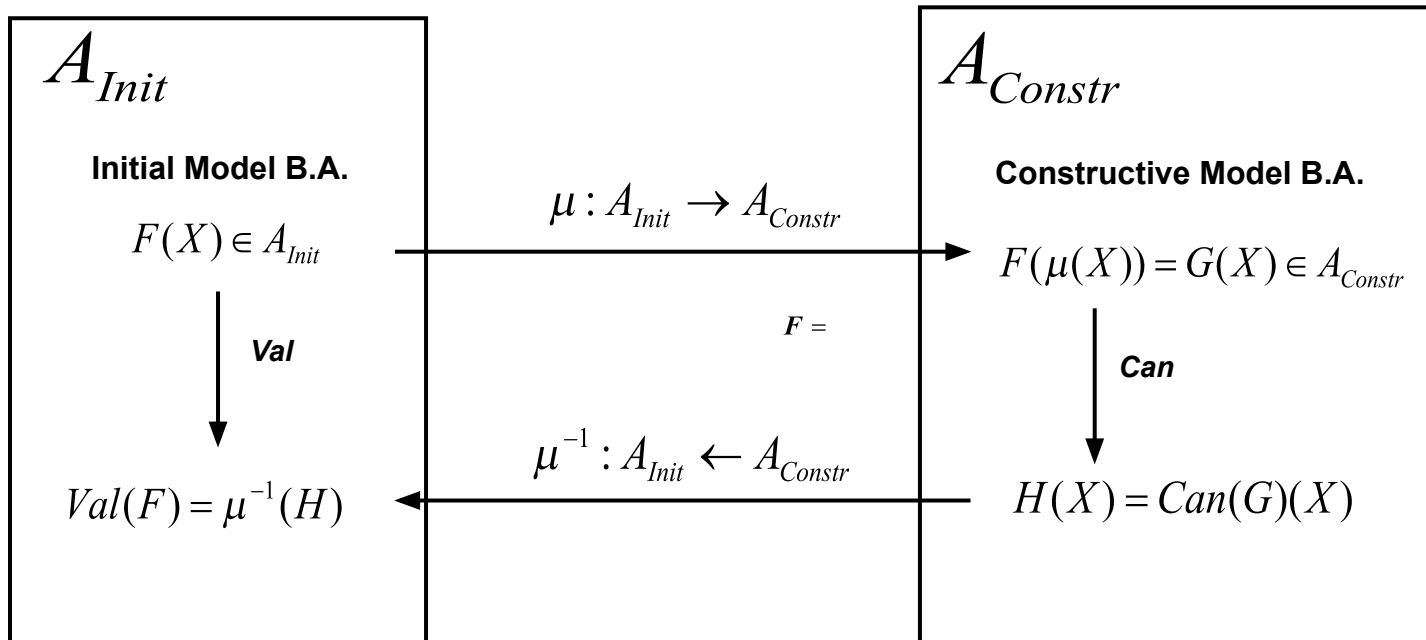
$$F(x_1, \dots, x_n) \equiv G(x_1, \dots, x_n) \Rightarrow Can(F)(x_1, \dots, x_n) = Can(G)(x_1, \dots, x_n) \quad (2)$$

Sign « $\equiv$ » means syntax equality (equality in free terms algebra)



Traditional logical canonical forms are **PDFN, PCNF**.  
 We shall use **Recursive Normal Form (RNF)**.

### Paradigm of Algebraic Computations



$$Val(F(X)) = \mu^{-1}(Can(F(\mu(X))))$$

(3)

## **Method of Inheritance**

**In algebraic computations are used:**

**Classical (axiomatically or constructively defined) algebraic structures,**

**Algebraic structures, defined by designer**

**Logical Designing of algebraic computations consists in defining of  
hierarchy of inheritance of MAS**

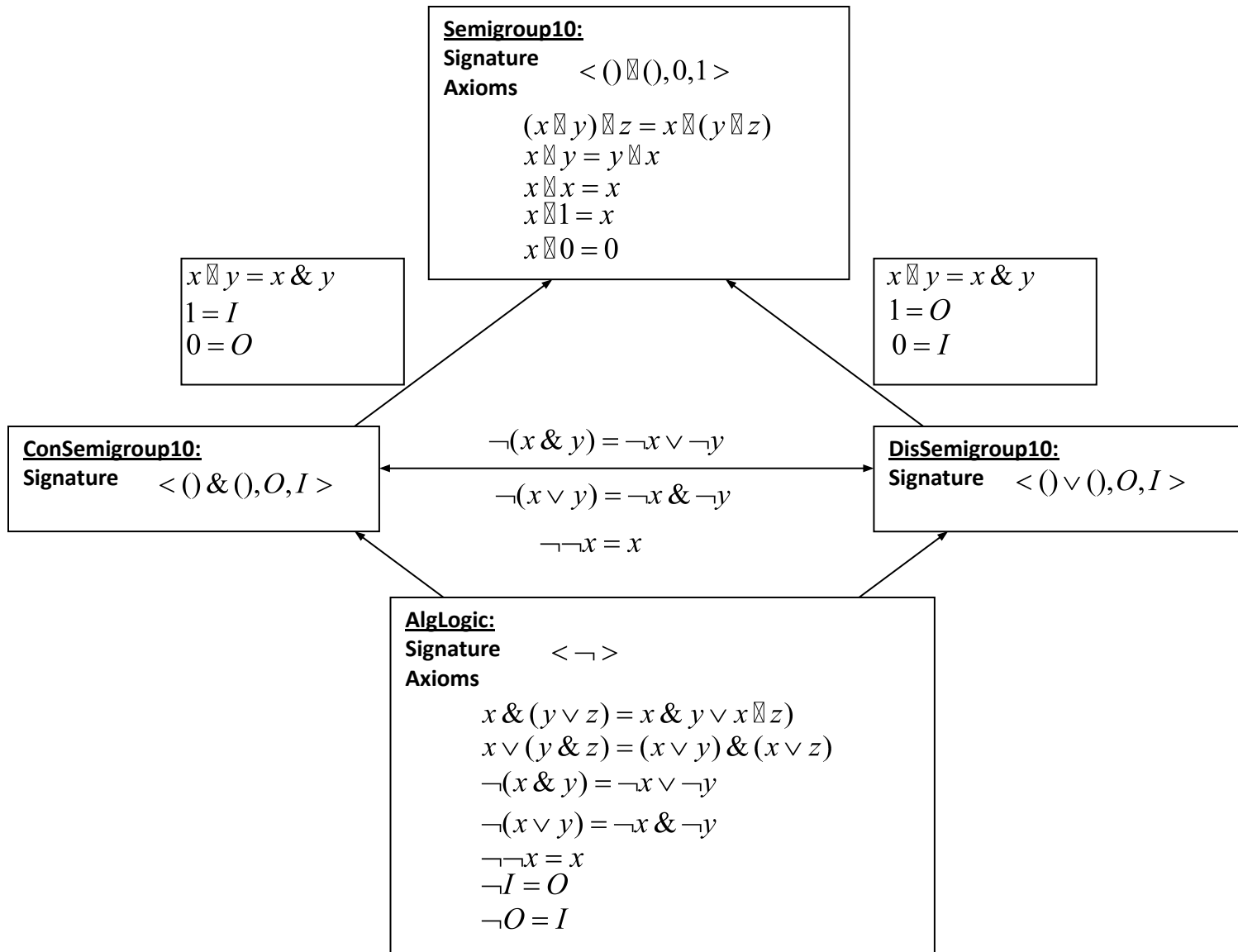


Рис. 3. Diagram of Inheritance for Boolean Algebra



**Axiomatic descriptions** plays constructive role. Its defines signatures, computations with constants.

**Inheritance** is used for definitions of algebraic operations interpreters with constants

SGOperation := rs(a)

{  
     $a^{\circ}1 = a, 1^{\circ}a = a,$   
     $a^{\circ}0 = 0, 0^{\circ}a = 0$   
};

Dis := rs(a)

{  
     $a \vee 0 = a, 0 \vee a = a,$   
     $a \vee 1 = 0, 1 \vee a = 1$   
};

Con := rs(a)

{  
     $A \& 1 = a, 1 \& a = a,$   
     $A \& 0 = 0, 0 \& a = 0$   
};

## Abstract Algorithms

**Inheritance** is used for descriptions of algorithms on abstract level (independently of algebras support).

Example: **Euclid's Algorithm** (abstract **Euclidian ring**).

## Derived logical operations

Imp := rs(a, b)

{

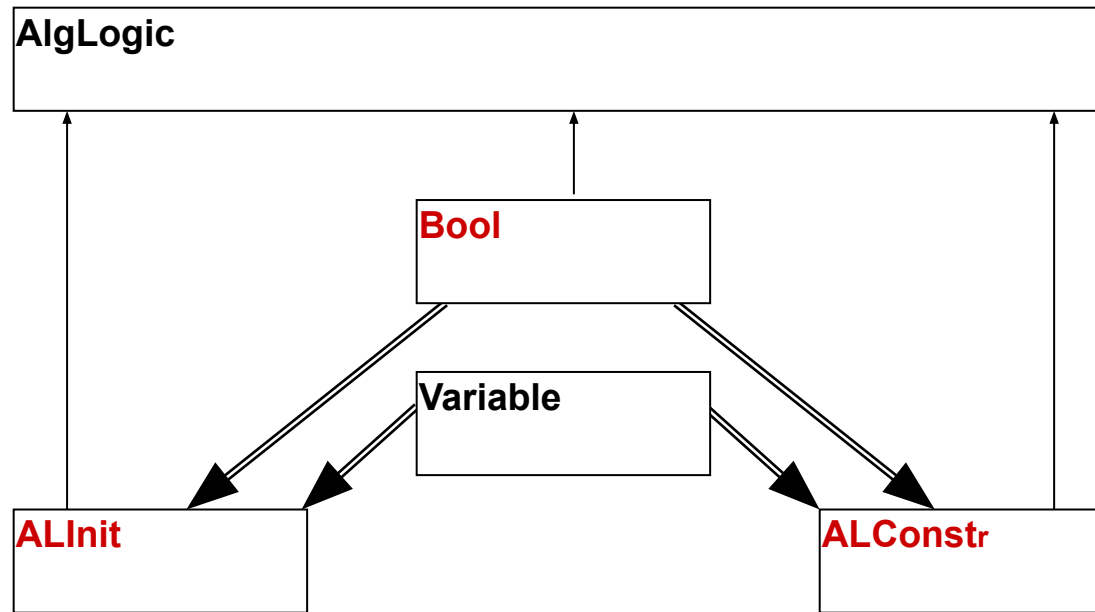
$a \rightarrow b = \neg a \vee b$

}

## Method of Extensions

Axiomatic definition of sort **AlgLogic** is initial for further development.

Three constructive models of **AlgLogic**:



Initial diagram of Method of extensions for Boolean algebra.

### Important:

Constructive definition of algebra **A** consists in definitions

- of its support **Sup(A)**,
- interpreters of its operations

## Bool $\mathbb{H}$ Variable – base sorts.

### Definition

Let  $\text{Bool} = \{O, I\}$ ,  $\text{Variable} = \{x_1, x_2, \dots\}$ .

Define sequence  $A^{(0)} \subset A^{(1)} \subset \dots \subset A^{(j)} \subset A^{(j+1)} \subset \dots$

1.  $A^{(0)} = \text{Bool}$ ,

2.  $A^{(j+1)} = \{F : F = L(F^{(j)}, G^{(j)}, x_{j+1})\}$ ,

where  $F^{(j)}, G^{(j)} \in A^{(j)}$ ,

$L$ -special functional symbol, (constructor of  $\text{ALConstr}$ ).

We use notation  $A(x)$  for algebra

$$A(x) = \{F : F = L(F_1, G_1, x), F_1, G_1 \in A, x \notin A\}$$

Elements of  $A(x)$  has the view  $L(F, G, x)$ .

Selectors :

$$\text{Arg}_1(L(F, G, x)) = F, \text{Arg}_2(L(F, G, x)) = G, \text{Var}(L(F, G, x)) = x$$

3. On *ALConstr* relations of inclusions are holds:

$$A \Rightarrow A(x)$$

$$L(F, F, x) = F$$

$$\text{Variable} \Rightarrow A(x)$$

$$L(I, O, x) = x$$

Algebra  $A_{Constr}$  is direct limit of increasing sequence  $A^{(j)}$ :

$$A_{Constr} = \bigcup_{j=0}^{\infty} A^{(j)}$$

Define for  $A_{Constr}$  logical operations. Let

$$L(A_1, B_1, x), L(A_2, B_2, x) \in A(\mathbf{x}), A_1, B_1, A_2, B_2 \in A.$$

Base rules:

$$L(A_1, B_1, x) \vee L(A_2, B_2, x) = L(A_1 \vee A_2, B_1 \vee B_2, x), \quad (10)$$

$$L(A_1, B_1, x) \& L(A_2, B_2, x) = L(A_1 \& A_2, B_1 \& B_2, x), \quad (11)$$

$$\neg L(A_1, B_1, x) = L(\neg A_1, \neg B_1, x). \quad (12)$$

Rules (10) - (12) are base for operations on

*ALConstr*

### **Additional Rules (partial cases):**

In the rule (10) suppose  $A_1 = B_1$ . then obtain:

$$\mathbf{A_1} \vee L(A_2, B_2, x) = L(A_1, A_1, x) \vee L(A_2, B_2, x) = L(A_1 \vee A_2, A_1 \vee B_2, x).$$

**This equality defines the conditional rewriting rule**

$$\text{Var}(A_1) < x \square A_1 \vee L(A_2, B_2, x) = L(A_1 \vee A_2, A_1 \vee B_2, x). \quad (10a)$$

**Analogously**

$$L(A_1, B_1, x) \vee \mathbf{A_2} = L(A_1, B_1, x) \vee L(A_2, A_2, x) = L(A_1 \vee A_2, B_1 \vee A_2, x).$$

$$\text{Var}(A_2) < x \square L(A_1, B_1, x) \vee A_2 = L(A_1 \vee A_2, B_1 \vee A_2, x) \quad (10b)$$

**Rules (10), (10a), (10b) forms rewriting rules system of interpreter of disjunction for operands of the form  $L(A, B, x)$ .**

**Dis := rs(A1, B1, A2, B2, x)**

**{**  
    **L(A1, B1, x) ∨ L(A2, B2, x) = L(A1 ∨ A2, B1 ∨ B2, x),**  
    **Var(A1) < x □ A1 ∨ L(A2, B2, x) = L(A1 ∨ A2, A1 ∨ B2, x),**  
    **Var(A2) < x □ L(A1, B1, x) ∨ A2 = L(A1 ∨ A2, B1 ∨ A2, x)**  
**};**

**Con := rs(A1, B1, A2, B2, x)**

**{**  
    **L(A1, B1, x) & L(A2, B2, x) = L(A1 & A2, B1 & B2, x),**  
    **Var(A1) < x □ A1 & L(A2, B2, x) = L(A1 & A2, A1 & B2, x),**  
    **Var(A2) < x □ L(A1, B1, x) & A2 = L(A1 & A2, B1 & A2, x)**  
**};**

**Not := rs(A1, B1, x)**

**{**  
    **¬L(A1, B1, x) = L(¬A1, ¬B1, x)**  
**};**



## Main Result of Extension's Method

Definition of algebra  $ALConstr$  as direct limit of increasing sequence of algebras in issue gives the algorithm of synthesis of interpreters for algebraic operations

Full rewriting rules systems for logical operations: Disjunction

Dis := rs(a, A1, B1, A2, B2, x)

{

$a \vee 0 = a, \quad 0 \vee a = a, \quad // \text{Inherited}$

$a \vee I = 0, \quad I \vee a = I,$

$L(A_1, B_1, x) \vee L(A_2, B_2, x) = L(A_1 \vee A_2, B_1 \vee B_2, x), \quad // \text{base}$

$\text{Var}(A_1) < x \square A_1 \vee L(A_2, B_2, x) = L(A_1 \vee A_2, A_1 \vee B_2, x), \quad // \text{additional}$

$\text{Var}(A_2) < x \square L(A_1, B_1, x) \vee A_2 = L(A_1 \vee A_2, B_1 \vee A_2, x)$

};

## Method of morphisms

### From diagram 1

Computations in  $A_{Init}$  defined by isomorphisms

$$\mu: A_{Init} \rightarrow A_{Constr}, \quad \mu^{-1}: A_{Constr} \rightarrow A_{Init}$$

For our considerations  $\mu$  defined on Variable  $\subset A_{Init}$ :

$$x = L(I, O, x) \tag{13}$$

**Inverse isomorphism**  $\mu^{-1}$  defined by:

$$L(F, G, x) = x \& F \vee \neg x \& G \quad // \text{RNF} \tag{14}$$

Base rules (10)-(12) can be obtained from (14) by equivalent transformations

## Direct isomorphism

ALInitToConstr := rs(x)

{

IsVar(x)  $\rightarrow$  x = L(I, O, x)

};

**Inverse isomorphism** (with some additional rules):

ALConstToInit := rs(x, A, B)

{

L(I, O, x) = x,

L(O, I, x) =  $\neg$ x,

L(A, O, x) = x&A,

L(O, B, x) =  $\neg$ x&B,

L(A, B, x) = x&A  $\vee$   $\neg$ x&B

};

## Verification of interpreters of algebraic computations

Axioms from axiomatic definition of algebra  $A_{Abstract}$  can be used for verification of its constructive model  $A_{Constr}$ .

Axioms of  $A_{Abstract}$  are identities of  $A_{Constr}$ .

The proof may be obtained by mathematical induction for index  $j$  of sequence

$$A^{(0)} \subset A^{(1)} \subset \dots \subset A^{(j)} \subset A^{(j+1)} \subset \dots$$

## Example

De Morgan's Law:

$$\neg(x \& y) = \neg x \vee \neg y \quad (1)$$

Let

$$x = L(A1, B1, u), \quad y = L(A2, B2, u).$$

The result of substituting in (1):

$$\neg(L(A1, B1, u) \& L(A2, B2, u)) = \neg L(A1, B1, u) \vee \neg L(A2, B2, u) \quad (2)$$

After computing of left and right hand sides using wrt Dis, Con and Not we obtain:

$$L(\neg(A1 \& A2), \neg(B1 \& B2), u) = L(\neg A1 \vee \neg A2, \neg B1 \vee \neg B2, u) \quad (3)$$

If to equate arguments left and right hand sides we obtain :

$$\neg(A1 \& A2) = \neg A1 \vee \neg A2, \quad \neg(B1 \& B2) = \neg B1 \vee \neg B2 \quad (4)$$

Satisfiability of (1) reduces for satisfiability of (4).

So to provide induction's base we must check (1) on base algebra Bool.

If all additional rules are correct the proof of correctness of (1) is over.

If all axioms of  $A_{Abstract}$  are correct, wrt Dis, Con and Not are correct.

## Литература

1. Lvov M., Kuprienko A., Volkov V. Applied Computer Support of Mathematical Training Proc. of Internal Work Shop in Computer Algebra Applications, Kiev. – 1993. – pp. 25-26.
2. Lvov M. AIST: Applied Computer Algebra System Proc. of ICSTE'93. Kiev. – pp. 25-26.
3. Львов М.С. Терм VII – шкільна система комп'ютерної алгебри Комп'ютер у школі та сім'ї. – 2004. – №7.- С. 27-30.
4. М.Львов. Концепция информационной поддержки учебного процесса и ее реализация в педагогических программных средах. Управляющие системы и машины.- 2009.-№2 (в печати).
5. Львов М.С. Синтез інтерпретаторів алгебраїчних операцій в розширеннях багатосортних алгебр (підготовлено до друку)
6. Львов М.С. Верифікація інтерпретаторів алгебраїчних операцій в розширеннях багатосортних алгебр (підготовлено до друку)
7. М.С.Львов. Метод спадкування при реалізації алгебраїчних обчислень в математичних системах навчального призначення (підготовлено до друку)
  - Львов М.С. Метод морфізмів реалізації алгебраїчних обчислень в математичних системах навчального призначення (підготовлено до друку)
1. Goguen J., Meseguer J. Ordered-Sorted Algebra I: Partial and Overloaded Operations. Errors and Inheritance. SRI International, Computer Science Lab., 1987.
2. Песчаненко В.С. Розширення стандартних модулів системи алгебраїчного програмування APS для використання у системах навчального призначення // Науковий часопис НПУ імені М.П. Драгоманова Серія №2. Комп'ютерно-орієнтовані системи навчання: Зб. наук. Пр./Редкол.- К.:НПУ ім.М.П.Драгоманова, - №3 (10), 2005. - С.206-215.
3. Песчаненко В.С. Об одном подходе к проектированию алгебраических типов данных // Проблемы програмування. - 2006.- №2-3.-С. 626-634.
4. Песчаненко В.С. Использование системы алгебраического программирования APS для построения систем поддержки изучения алгебры в школе // Управляющие системы и машины.- 2006.- №4. - С. 86-94.
5. Letichevsky A., Kapitonova J., Volkov V., Chugajenko A., Chomenko V. Algebraic programming system APS (user manual) Glushkov Institute of Cybernetics, National Acad. of Sciences of Ukraine, Kiev, Ukraine, 1998.
6. Капитонова Ю.В., Летичевский А.А., Волков В.А. Дедуктивные средства системы алгебраического программирования, Кибернетика и системный анализ, 1, 2000, 17-35.
7. Kapitonova J., Letichevsky A., Lvov M., Volkov V. Tools for solving problems in the scope of algebraic programming. Lectures Notes in Computer Sciences. –№ 958. – 1995. – pp. 31-46.
  - Львов М.С. Основные принципы построения педагогических программных средств поддержки практических занятий. Управляющие системы и машины.- 2006.-№6. с. 70-75.