Models of alternative selection

Abstract

Models of a number of eliminations in conditions have been presented. In particular, the models of a number of eliminations in the form of a block diagram of the algorithm, the instructions if() and switch() of algorithmic programming languages, systems of algorithmic algebras and its modifications, as well as two models of algebraic algorithms have been constructed.

Keywords: selection condition, block diagram, system of algorithmic algebras, algebra of algorithms, model.

1. Introduction

Modern systems, including automation and control are complex. Their operation is implemented in accordance with their algorithms. Typically they have a lot of functionality. The combination of different modes of operation requires design of branching algorithms. The selection of alternative modes of operation is based on the fulfillment or non-fulfillment of certain conditions. There are many cases when the fulfillment of a number of existing conditions is required.

The article presents various methods of representation of a number of conditions. In particular, it illustrates the use of block-diagram method, conventional instructions of modern languages of object programming and algebraic methods. Due to the fact that the algebraic methods provide the most accurate and compact representation of certain conditions, they are considered in more detail.

The known algebraic methods of representation of algorithms is a system of algorithmic algebras and its modification [1, 2], and algebra of algorithms [3 - 6].

2. Representation of a number of conditions by block diagram

Block diagram method is represented in the form of diamond. They are called conditional blocks that have one input and two outputs. The condition is written in the middle of the block. That output, which occurs when the condition fails is defined as number 0 or the word "No".

![Fig. 1. Block diagram of abstract algorithm with a number of conditions](image)

Fig. 1 represents an abstract algorithm with a number of conventional blocks. The fulfillment of conditions умов \( u_0, u_1, u_2, \ldots, u_{n-1} \) is accompanied by fulfillment of operational blocks \( F_0, F_1, F_2, \ldots, F_{n-1} \), respectively. Failure to fulfill any of the conditions leads to the selection of operational block \( Y \).

3. Representation of a number of conditions by algorithmic languages

In algorithmic programming languages, that are the languages of assembly, procedural or object programming of computing machines and systems, the conditions are described by the instruction if().

Represented in Fig.1 block diagram of the algorithm in a very close form to the language code of object programming can be written in the following form:

```java
if(u0) F0
   else if(u1) F1
   else if(u2) F2
      \ldots
   else if(un-1) Fn-1
   else Y;
```

where if is a conditional operator “if”, a else means “others”.

It is well known that the languages of object programming, for example, C# language, use operator switch () [7]. In principle, this operator can be replaced by operator if(). However, in practice of programming, as a rule, when there are three or more consecutive executable with mutually exclusive conditions they use the operator switch (). Its use is due to getting clearer and relatively easily generated and readable code.

With regard to the shown in Fig.1 block diagram of the algorithm, the instruction switch, for example, can be written in this form:

```java
using System;
public class Instrukcja
{
    static void Main( string [] t)
    {
        switch (t[0])
        {
            case "u0":
                F0
                break;
            case "u1":
                F1
                break;
            case "u2":
                F2
                break;
            \ldots
            case "un-1":
                Fn-1
                break;
            default:
                Y;
        }
    }
}
```

The above instructions describes: the access (using) to the system namespace System; public access (public) to a class (class), which is called Instrukcja; static (static) and void (void) value method named Main, which has one input parameter, that is the table ([] called t, in whose cells text (string) values are stored; break is the identifier of fulfillment completion; default is the identifier of completion of the last component.

Actually, the instructions switch is used the most frequently in programming.
4. Model of the system of algorithmic algebras

The system of algorithmic algebras and its modification [1, 2] to describe the conditions gives the operation of alternative that has the form \((u_1 A B)\). The fulfillment of the condition leads to the selection of the operator \(A\), and failure – to the operator \(B\).

Applying the operation of alternatives to the description of the algorithm Fig.1, we obtain the following formula

\[
([u_0] F_0 ([u_1] F_1 ([u_2] F_2 \ldots ([u_{n-1}] F_{n-1} Y) \ldots )).
\]

5. Model of algebra of algorithms

Modified algebra of algorithms contains the axiom of generalization of the elimination operation of functional uniterm \(F(x)\), which depends on variable \(x\), that runs an infinite number of values for this variable. The variable takes the value from the sequential field of values.

The result of this generalization is the operation of cyclic elimination

\[
F(i); F(j); F(k); \ldots ; u_0; u_1; u_2; \ldots ; u_n \Rightarrow \sum F(x); Y; u_0
\]

Since the operation of elimination can be realized by operator \(if()\), and the operation of multi-elimination can be realized by operator \(switch()\), then the operation of cyclic elimination

\[
\sum F(x); Y; u_0? = x \in a; b; e; \ldots
\]

which is the generalization of the operation of elimination can be seen as theoretical generalization of these operators.

A similar situation exists in mathematical logic. There is the operation of disjunction and quantifier operation for "some." For example, the use of existential quantifier to variable \(x\) of predicate \(P(x)\) gives the formula \(\sum x P(x)\), which, in general, for infinite number of values of variable \(x(\{e_0, e_1, e_2, \ldots \})\) can be viewed as the generalization the operation of disjunction in infinite number of values of variable \(x\). The operation of cyclic elimination

\[
\sum F(a); (b); (c); \ldots ; u_0; u_1; u_2\ldots
\]

in the case of the deployment of cyclic elimination by the operations of ordinary (non-cyclic) elimination, this operation for a finite number of values of variable \(x\), for example, for three values of the variables

\[
x \in a; b; e; \ldots
\]

will deploy into the following expression

\[
F(a); F(b); F(c); Y; u_0?; u_0?; u_0? .
\]

In case of cyclic elimination in such formula

\[
\sum F(x); Y; u_0? ,
\]

that from the previous cyclic elimination differs by rearrangement of uniterms \(F(x)\) and \(Y\) in the operation of elimination, the deployed cyclic elimination for the above mentioned three values of variable \(x\), will have the form

\[
Y; F(c); u_0?; F(b); u_0?; F(a); u_0?.
\]

The resulting deployed formula of cyclic elimination differs from the previous deployed formula of cyclic elimination in the selection by inverse values of conditional uniterms \(u_a, u_b\) and \(u_c\) of components \(F(a), F(b)\) and \(F(c)\), respectively.

Considering one important case, and in fact it is in receiving answers to the question: whether formulas of cyclic elimination, that are the expressions

\[
\sum F(x); Y; u_0? \quad \sum F(x); Y; u_0? ,
\]

provide the opportunity to obtain the same functional uniterms when fulfilling different conditional uniterms.

**Theorem.** Formula of cyclic elimination provides the possibility to obtain the same functional uniterms if there is paralleling of conditional uniterms.

**Proof.** Considering the formula

\[
\sum F(x); Y; u_0? ,
\]

which, for example, for

\[
x \in a; b; e; \ldots
\]

deploys to the expression

\[
F(a); F(b); F(c); Y; u_0?; u_0?; u_0? .
\]

Let functional uniterms \(F(a)\) and \(F(b)\) are the same, i.e. \(F(a)=F(b)\). They are obtained by fulfilling conditional uniterms \(u_a\) and \(u_b\) respectively. Replacing \(F(b)\) to \(F(a)\) in the last elimination we get the formula

\[
F(a); F(a); F(c); Y; u_0?; u_0?; u_0? .
\]

On the basis of axiom

\[
\sum F(x); Y; u_0? \quad \sum F(x); Y; u_0? ;
\]

dropping in question signs in conditional uniterms, we replace the first elimination by paralleling of sequences, and we obtain the expression

\[
\sum u_a; u_b
\]

We again apply the axiom of relationships between the operation of elimination and paralleling and sequencing, we obtain the formula

\[
\sum u_a; u_b
\]

On the basis of axiom

\[
\sum F(a); F(a); F(c); Y; u_0; u_0.
\]

We deduce the expression

\[
\sum u_a; u_b
\]

from the given formula we deduce

\[
\sum u_a; u_b
\]
Applying the property of removal of variable operation of sequencing to the received expression

we obtain

This formula contains the expression

Taking into account the fact that here we have the axiom

and the axiom

from which, replacing \( u_1 \) and \( u_2 \) into \( u \) and taking into account that

and

we get

Substituting \( u \) by \( u_b \) in it and we substitute the received formula of paralleling instead of 1 in

Substituting it into the formula

instead of non-inverting \( u \) and doing the obvious and elementary transformations we obtain the formula

Substituting it into the previous expression we get the formula

We have this expression in it

which is replaced by the formula

Replacing the operation of paralleling with it, we get the formula

Applying to it the axiom of relationships between elimination and paralleling of sequences, we obtain the expression

Replacing

by

finally we get

The theorem is proved the same way for the second case of cyclic elimination. The theorem is proved.

6. Conclusions

The model of description of a number of eliminations in the form of a block diagram is the most visible of all the other models submitted. Models of algorithmic programming languages, compared with the model in the form of block diagram are less visible but more compact. The system of algorithmic algebras and its modification provides construction of the model, which is less visible for all previous models but it is the most compact among them. Algebra of algorithms provides the construction of two models. The first model is in the form of multiple eliminations. The second model is in the form of the operation of cyclic elimination, which is the most compact among all models.

7. References


Received: 01.12.2014     Paper reviewed     Accepted: 02.03.2015

Prof. Ph.D. eng. Volodymyr OVSYAK

He specializes in theoretical and applied computer science, theory of algorithms, programming, information systems, mathematical modeling of systems, computer simulation and mathematical modeling. Currently he works as an professor in Kielce University of Technology.

e-mail: ovsyak@rambler.ru

Ph.D. eng. Oleksandr OVSYAK

He specializes in theoretical and applied computer science, theory of algorithms, programming, information systems, mathematical modeling of systems, computer simulation and mathematical modeling. He works as an associate professor in National University of Culture and Arts in Lviv, Ukraine.

e-mail: ovsjak@ukr.net