Annales Univ. Sci. Budapest., Sect. Comp. 31 (2009) 185-202

MANY-VALUED LOGIC, MAPPINGS, ICF GRAPHS, NORMAL FORMS

K. Pásztor Varga (Budapest, Hungary)

M. Várterész (Debrecen, Hungary)

Abstract. The role of (canonical) conjunctive and disjunctive normal forms ((C)CNF, (C)DNF) is very important both in classical and many-valued logic. Normal forms have been proved as fundamental tools in automated theorem proving, in electrical engineering and, in the investigation of the complexity of logical mappings $\{true, false\}^m \to \{true, false\}$ as well. The iterative canonical form (ICF) graph introduced in [10] is a successful approach for that. Later, this idea was applied for biological problems in [8, 9]. One extension of this idea to many-valued case was developed for logical design in [4]. There exist some other interesting works at laboratory level also.[1] In this paper, we give a possible method based on the lattice diagram of an n-valued logical mapping $E_n^m \to E_n$ to find the ICF graph. Using ICF graphs we construct describing formulae as well. In order to prepare this description we give a short (fitting to our aims) summary of the many-valued logic.

1. Introduction

The classical propositional logic can be considered as a lattice for any ordering of the two truth values. The lattice operations are the operations $\min(x, y)$ and $\max(x, y)$. These operations together with the negation operation (as operation generating the complement) are functionally complete. Normal forms ((C)CNF, (C)DNF) containing only these three operations are the most important means in the description of logical mappings $\{true, false\}^m \to \{true, false\}$. A conjunction of j variables ($j < m$) represents the minimum of all the points in

$\{true, false\}^m$ having *true* values at the position of fixed j variables - i.e. it is an $(m-j)$ dimensional Boolean cube. Basing on the ICF graph of a given mapping it is possible to construct a formula over $\{\neg, \wedge, \vee\}$ which describes this mapping [10]. There exist canonical normal forms in the many-valued logic. Let E_n $(n \geqslant 2)$ be the set $\{0, 1, 2, \ldots, n-1\}$. The elements of the set E_n are our truth values in the n-valued logic. A conjunction of j variables $(j < m)$ represents the minimum of all the points from E_n^m having truth values $e_1, e_2, ..., e_j$ at the position of the j variables - that is an $(m-j)$ dimensional E-cube. However, the set of all points having positive truth values at every position can not be the result of any simplification. Consequently, these E-cubes are represented by their least points and not by conjunctions. Then the construction of the ICF graph is possible using the m-dimensional lattice diagram. But to obtain a describing formula is an other question. We show a possible way to construct the ICF graph and its using to compute the complexity of the original mapping. Moreover, we show an attempt to construct the describing formula.

2. The many-valued logics

The classical propositional logic is defined by its syntax and semantics. The syntax is given by a pair $\langle Prop, Con\rangle$, where $Prop$ is the set of propositional variables (letters), Con is the set of connectives (for example $Con = \{\neg, \wedge, \vee\}$). Well-formed formulas can built by structural induction on Con in the usual way. Meaning is assigned to well-formed formulas by a semantics $\langle\{0,1\}, I, A(Con)\rangle$, where $\{0,1\}$ is the set of truth values (*true* is signed by 1 and *false* by 0), $I : Prop \to \{0,1\}$ is called interpretation and $A(Con)$ are operations on $\{0,1\}$ of suitable arity signed by members of Con. (The set $\{A(\neg), A(\wedge), A(\vee)\}$ is functionally complete.)

Like the classical logic, many-valued propositional logics is also defined by its description language, the set of truth values and the operations on it. In accordance with classical logic, the description language is given by a pair $\langle Prop, Con\rangle$, where $Prop$ is the set of variables, $Con = \{C_1, C_2, \ldots, C_k\}$ is the set of connectives. At the same time, a many-valued logic manages a truth value set E with the cardinality larger than the usual two. The semantics of a many-valued logic even now is a triple $\langle E, I, A(Con)\rangle$, where $I : Prop \to E$ is an interpretation, $A(Con) = \{A(C_1), A(C_2), \ldots, A(C_k)\}$ are operations on E of suitable arity.

In the description language, Con contains logical connectives, $A(Con)$ contains operations on E associated to that connectives. In principle the extensions of the classical operations are the members of $A(Con)$.

By way of example, we present the operations conjunction and disjunction in

Lukasiewicz

$\wedge$	t	f	i
t	t	f	i
f	f	f	f
i	i	f	i

Bochvar

$\wedge$	t	f	m
t	t	f	m
f	f	f	m
m	m	m	m

Kleene

$\wedge$	t	f	u
t	t	f	u
f	f	f	f
u	u	f	u

Lukasiewicz

$\vee$	t	f	i
t	t	t	t
f	t	f	i
i	t	i	i

Bochvar

$\vee$	t	f	m
t	t	t	m
f	t	f	m
m	m	m	m

Kleene

$\vee$	t	f	u
t	t	t	t
f	t	f	u
u	t	u	u

Table 1. Operations in some Three-Valued Logics

the most famous three-valued logics (from [1]) in the Table 1. Here we use the conventional notation for the truth values: *true* is signed by t, *false* by f and the third truth value stands for

- poss-i-ble, not yet determined for Lukasiewicz,
- m-eaningless (paradoxical) for Bochvar,
- u-ndefined for Kleene.

Henceforth by standard simplification of notation, we use the same symbol to refer both to a connective and to the operation on the truth values associated to that connective.

3. Designation of truth values

In this paper we deal with so called n-valued logic. Let E_n $(n \geqslant 2)$ be the set $\{0, 1, 2, \ldots, n-1\}$. The members of the set E_n represent our truth values. To define the notion of semantical consequence the designated values (from [15]) are important. Let $0 \leqslant S < n-1$. $\{0, 1, \ldots, S\}$ are *non-designated truth values*, and $\{S+1, \ldots, n-1\}$ the *designated* ones.

Definition 3.1. *In an interpretation I, a formula $\mathcal{F}$ is said to be:*

- *S-assertable, if its truth value in I is designated;*
- *S-unassertable, if its truth value in I is non-designated.*

Definition 3.2. *A formula $\mathcal{F}$ is said to be:*

- *S-tautology, if it is S-assertable in all interpretations,*
- *S-satisfiable, if there is an interpretation, where $\mathcal{F}$ is S-assertable.*

Definition 3.3. *A formula $\mathcal{G}$ is a semantical S-consequence of formulae $\mathcal{F}_1, \ldots, \mathcal{F}_m$ (denoted as $\mathcal{F}_1, \ldots, \mathcal{F}_m \models_S \mathcal{G}$) if for any interpretation in which every formula $\mathcal{F}_i$ ($1 \leqslant i \leqslant m$) is S-assertable, $\mathcal{G}$ is also S-assertable with at least the same truth value as the maximum of the truth values of formulae $\mathcal{F}_1, \ldots, \mathcal{F}_m$ in the underlying interpretation.*

In classical logic $\mathcal{F}_1, \ldots, \mathcal{F}_m \models \mathcal{G}$ iff $\{\mathcal{F}_1, \ldots, \mathcal{F}_m, \neg\mathcal{G}\}$ is unsatisfiable. This decision problem necessitates a negation operation having the suitable result in many-valued logic: a negation of an S-assertable formula is S-unassertable and vice versa. For example Rosser in [15] defined such a negation.

4. Operations in the n-valued logic

We have already seen, the operations in the n-valued logic must be the extensions of the operations of the classical two-valued logic. Some possible extension of classical logical operations:

- n-valued conjunction $x_1 \wedge x_2 \rightleftharpoons \min(x_1, x_2)$,
- n-valued disjunction $x_1 \vee x_2 \rightleftharpoons \max(x_1, x_2)$,
- Lukasiewicz negation $\tilde{\neg}x \rightleftharpoons n - 1 - x$,
- Post negation $\bar{\neg}x \rightleftharpoons (x + 1) \bmod n$,
- Lukasiewicz implication $x_1 \tilde{\supset} x_2 \rightleftharpoons \begin{cases} n-1 & \text{if } x_1 \leqslant x_2 \\ (n-1) - x_1 + x_2 & \text{if } x_1 > x_2 \end{cases}$
- Post implication $x_1 \bar{\supset} x_2 \rightleftharpoons \begin{cases} n-1 & \text{if } x_1 \leqslant x_2 \\ x_2 & \text{if } x_1 > x_2,\ S \leqslant x_1 \\ (n-1) - x_1 + x_2 & \text{if } x_1 > x_2,\ S > x_1 \end{cases}$

There are some problems with some of these n-valued operators.

- The Lukasiewicz and Post negations of a designated truth value is not non-designated in every case.

  If $n = 3$, $S = 1$, then the truth value 1 is non-designated. In the same case, Lukasiewicz negation of the truth value 1 has

  $$\tilde{\neg}1 = 3 - 1 - 1 = 1$$

  truth value, so it is non-designated too.

- In order to be able to formulate the decision problem in the many-valued logics it would be necessary such a negation ($\neg$) and implication ($\supset$) that $x_1 \supset x_2$ is the same as $\neg x_1 \vee x_2$. (The modus ponens requires such an implication.)

  There are cases, when x_1 and the Lukasiewicz implication $x_1 \tilde{\supset} x_2$ are designated, but x_2 is not. For example if $n = 3, S = 0, x_1 = 1, x_2 = 0$, then

$$x_1 \tilde{\supset} x_2 = (n-1) - x_1 + x_2 = 1.$$

5. Functionally completeness

Any function $f : E_n^m \to E_n$ can be considered as an n-valued operation. A set X of n-valued operation is said to be *functionally complete* iff every operation is definable by means of operations in the set X.

Theorem 5.1. *The operation set $\{\neg, \wedge, \vee\}$ over E_2 is functionally complete.*

Now we introduce two unary operations in E_2, one of them is instead of $\neg$

$$j_0 x \rightleftharpoons \begin{cases} 1 & \text{if } x = 0, \\ 0 & \text{if } x = 1, \end{cases}$$

and the other is the identity

$$j_1 x \rightleftharpoons \begin{cases} 1 & \text{if } x = 1, \\ 0 & \text{if } x = 0. \end{cases}$$

Obviously, $\{j_0, j_1, \wedge, \vee\}$ is functionally complete over E_2.

Theorem 5.2. *Let $f : E_2^m \to E_2$ be an m-ary operation. Then it has a canonical disjunctive normal form:*

$$f(x_1, x_2, \ldots, x_m) = \bigvee_{(i_1, i_2, \ldots, i_m) \in E_2^m} \Big(\bigwedge_{k=1}^{m} j_{i_k} x_k \wedge f(i_1, i_2, \ldots, i_m) \Big).$$

Example 5.1. For example if $f : E_2^2 \to E_2$ is a binary operation, we get so that

$$\begin{aligned} f(x_1, x_2) = {} & (j_0 x_1 \wedge j_0 x_2 \wedge f(0,0)) \vee (j_0 x_1 \wedge j_1 x_2 \wedge f(0,1)) \vee \\ & \vee (j_1 x_1 \wedge j_0 x_2 \wedge f(1,0)) \vee (j_1 x_1 \wedge j_1 x_2 \wedge f(1,1)). \end{aligned}$$

i_1	i_2	$f(i_1,i_2)$	$j_{i_1}x_1 \wedge j_{i_2}x_2 \wedge f(i_1,i_2)$
0	0	$f(0,0)$	$j_0x_1 \wedge j_0x_2 \wedge f(0,0)$
0	1	$f(0,1)$	$j_0x_1 \wedge j_1x_2 \wedge f(0,1)$
1	0	$f(1,0)$	$j_1x_1 \wedge j_0x_2 \wedge f(1,0)$
1	1	$f(1,1)$	$j_1x_1 \wedge j_1x_2 \wedge f(1,1)$

Let us now $f(0,0)=0, f(0,1)=1, f(1,0)=0, f(1,1)=1$, that is with usual point-denotation $\{(00)0,(01)1,(10)0,(11)1\}$. For the sake of simplicity, in two-valued case, the complete elementary conjunctions belonging to the value 0 of function f do not appear in the normal forms, so the DNF of this operation is

$$f(x_1,x_2) = (j_0x_1 \wedge j_1x_2) \vee (j_1x_1 \wedge j_1x_2).$$

Resubstituting the usual signs ($\neg$ and ,,identity") and carrying out possible simplification we get

$$f(x_1,x_2) = (\neg x_1 \wedge x_2) \vee (x_1 \wedge x_2) = x_2.$$

Similarly, in the n-valued case for $i = 0,1,\ldots,n-1$ let be

$$j_ix \rightleftharpoons \begin{cases} n-1 & \text{if } x = i, \\ 0 & \text{if } x \neq i. \end{cases}$$

Theorem 5.3. *The operation set*

$$\{j_0, j_1, \ldots, j_{n-1}, \wedge, \vee\}$$

over E_n is functionally complete.

Theorem 5.4. *Let $f : E_n^m \to E_n$ be an m-ary operation. Then it has a CDNF:*

$$f(x_1,x_2,\ldots,x_m) = \bigvee_{(i_1,i_2,\ldots,i_m)\in E_n^m} \left(\bigwedge_{k=1}^{m} j_{i_k}x_k \wedge f(i_1,i_2,\ldots,i_m) \right).$$

Example 5.2. For example if $f : E_3^2 \to E_3$ is a binary operation, we get so that

$$\begin{aligned} f(x_1,x_2) = \\ &= (j_0x_1 \wedge j_0x_2 \wedge f(0,0)) \vee (j_0x_1 \wedge j_1x_2 \wedge f(0,1)) \vee (j_0x_1 \wedge j_2x_2 \wedge f(0,2)) \vee \\ &\vee (j_1x_1 \wedge j_0x_2 \wedge f(1,0)) \vee (j_1x_1 \wedge j_1x_2 \wedge f(1,1)) \vee (j_1x_1 \wedge j_2x_2 \wedge f(1,2)) \vee \\ &\vee (j_2x_1 \wedge j_0x_2 \wedge f(2,0)) \vee (j_2x_1 \wedge j_1x_2 \wedge f(2,1)) \vee (j_2x_1 \wedge j_2x_2 \wedge f(2,2)). \end{aligned}$$

i_1	i_2	$f(i_1,i_2)$	$j_{i_1}x_1 \wedge j_{i_2}x_2 \wedge f(i_1,i_2)$
0	0	$f(0,0)$	$j_0x_1 \wedge j_0x_2 \wedge f(0,0)$
0	1	$f(0,1)$	$j_0x_1 \wedge j_1x_2 \wedge f(0,1)$
0	2	$f(0,2)$	$j_0x_1 \wedge j_2x_2 \wedge f(0,2)$
1	0	$f(1,0)$	$j_1x_1 \wedge j_0x_2 \wedge f(1,0)$
1	1	$f(1,1)$	$j_1x_1 \wedge j_1x_2 \wedge f(1,1)$
1	2	$f(1,2)$	$j_1x_1 \wedge j_2x_2 \wedge f(1,2)$
2	0	$f(2,0)$	$j_2x_1 \wedge j_0x_2 \wedge f(2,0)$
2	1	$f(2,1)$	$j_2x_1 \wedge j_1x_2 \wedge f(2,1)$
2	2	$f(2,2)$	$j_2x_1 \wedge j_2x_2 \wedge f(2,2)$

Let be given an operation by point-denotation:

$$\{(00)0, (01)2, (02)1, (10)0, (11)2, (12)0, (20)0, (21)2, (22)0\}.$$

The complete elementary conjunctions belonging to value 0 of the operation f do not appear in the normal forms either:

$$f(x_1,x_2) = \\ = (j_0x_1 \wedge j_1x_2 \wedge 2) \vee (j_0x_1 \wedge j_2x_2 \wedge 1) \vee (j_1x_1 \wedge j_1x_2 \wedge 2) \vee (j_2x_1 \wedge j_1x_2 \wedge 2).$$

After simplification we get

$$f(x_1,x_2) = (j_1x_2 \wedge 2) \vee (j_0x_1 \wedge j_2x_2 \wedge 1).$$

Some pairs of syntax and semantics of the n-valued logics form special mathematical structures. Well known fact, that classical two-valued logic with connectives $Con = \{\neg, \wedge, \vee\}$ as a mathematical structure (the set $\{0,1\}$ with the operations $\{\neg, \wedge, \vee\}$) is a Boolean algebra. Now we will see that some n-valued logic as a mathematical structure is a Post algebra with the domain E_n ($= \{0,1,2,\ldots,n-1\}$) and some set of n-valued operations.

Definition 5.1. *By an n-valued Post algebra over E_n ($n \geqslant 2$) we mean an algebra*

$$\langle E_n, \{k_0,\ldots,k_{n-1}, j_0,\ldots,j_{n-1}, \wedge, \vee\}\rangle$$

where $\langle E_n, \{\wedge,\vee\}\rangle$ is a bounded distributive lattice with a zero 0 and a unit $n-1$, moreover, for each $i,l = 0,\ldots,n-1$ and $x,y \in E_n$,

(P1) *$x \vee y = y$, $x \wedge y = x$ for $x < y$,*

(P2) *if $i \neq l$, then $j_ix \wedge j_lx = 0$, $k_ix \vee k_lx = n-1$,*

(P3) $x = \wedge_{i=0}^{n-1}(k_i x \vee i)$, $x = \vee_{i=0}^{n-1}(j_i x \wedge i)$,

(P4) $0 = \wedge_{i=0}^{n-1} k_i x$, $n-1 = \vee_{i=0}^{n-1} j_i x$,

(P5) *if* $i \neq 0$, *then*
$x \vee (i-1) = i$ *implies* $x = i$, *and* $x \wedge i = i-1$ *implies* $x = i-1$.

Theorem 5.5.

$$P_2 \rightleftharpoons \langle E_2, \{\neg, \wedge, \vee\}\rangle$$

is a Boole algebra with the natural ordering of the set E_2.

Let be defined the operations $k_i (i = 0, 1, \ldots, n-1)$ as follows:

$$k_i x \rightleftharpoons \begin{cases} 0 & \text{if } x = i, \\ n-1 & \text{if } x \neq i. \end{cases}$$

The following theorem has been proved in [2].

Theorem 5.6.

$$P_n \rightleftharpoons \langle E_n, \{k_0, \ldots, k_{n-1}, j_0, j_1, \ldots, j_{n-1}, \wedge, \vee\}\rangle$$

is a Post algebra of order n *with the natural ordering of the set* E_n.

Now let be defined the ,,threshold" operation x^i according to [4]

$$x^i \rightleftharpoons \begin{cases} 0 & \text{if } x < i, \\ n-1 & \text{if } x \geqslant i. \end{cases}$$

By this operation it is possible to give a conjunction formula describing a set of points, the minimal point of which is the point given by the exponents.

Example 5.3. If a point in the 5-valued logic is (1232)2, then

$$\min(2, x^1, y^2, z^3, v^2) \text{ or } (2 \wedge x^1 \wedge y^2 \wedge z^3 \wedge v^2)$$

assigns the value 2 to (1232) and to all greater points.

Now let we define a negation-like operation which implements the set difference. Let $\mathcal{G}$ be a formula and let k be the assigned value in certain $I_{\mathcal{G}}^k$ interpretation. Further let $\mathcal{F}$ be another formula, where l is the value of $\mathcal{F}$ in the $I_{\mathcal{G}}^k$ interpretations. Let $I_{\mathcal{F}}^l \subseteq I_{\mathcal{G}}^k$. Then the formula, which has the value k in the interpretation $I_{\mathcal{G}}^k \setminus I_{\mathcal{F}}^l$ is exactly $\mathcal{G} \wedge \neg_k \mathcal{F}$, where

$$\neg_k \mathcal{F} \rightleftharpoons \begin{cases} 0 & \text{if } k \neq l, \\ n-1 & \text{if } k = l. \end{cases}$$

Example 5.4. Let $\mathcal{G} = (2 \wedge x^1 \wedge y^2 \wedge z^3 \wedge v^2)$ and $\mathcal{F} = (1 \wedge x^2 \wedge y^3 \wedge z^4 \wedge v^3)$, then $\mathcal{G} \wedge \neg_k \mathcal{F} = (2 \wedge j_1 x \wedge j_2 y \wedge j_3 z \wedge j_2 v)$.

6. Characterization of operations with lattice diagrams

Let be given the two-valued operation

$$\{(000)0, (001)1, (010)1, (100)0, (011)1, (101)0, (110)1, (111)0\}$$

by the lattice diagram of the space $\{0,1\}^3$ (Figure 1).

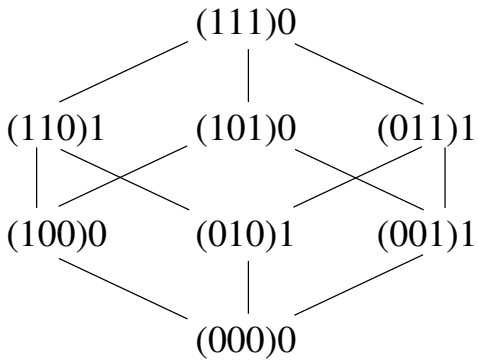

Figure 1. Lattice diagram of a two-valued operation

The ICF graph method for the two-valued case is an idea of Jakó [10]. As a result of some iterative transformations carried out on this graph (lattice diagram) we get some DNF pairs (D_i^1, D_i^2) containing only non negated variables, and the operation can be expressed by the formula

$$\bigvee_{i=1,2,\ldots,l} (D_i^1 \wedge \neg D_i^2).$$

The key idea to obtain the mentioned formula is the replacement of 2 (or 2^k) neighbour points by the least point in the ICF graph and by the elementary conjunction in which the variables correspond to the position where the 1 value appears in the least point. For example in the case of the points (010)1, (110)1 we get $x_2 \wedge \neg x_3$ from $\neg x_1 \wedge x_2 \wedge \neg x_3$ and $x_1 \wedge x_2 \wedge \neg x_3$. If the point-value is 0, then we can proceed in the same way, however the resulted formula should be negated.

An iteration step includes an α-expansion and a β-reduction. During an α-expansion we start with lowest 1-value points ((010)1 and (001)1) and we create the disjunction of conjunction(s) of non negated variables $(x_2 \vee x_3)$. We mark (cf. boldface) in the lattice diagram all points being greater than the starting

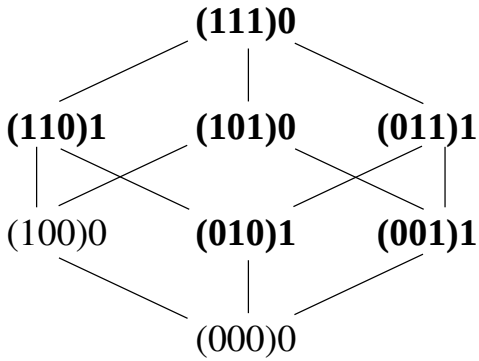

Figure 2. α-expansion of lattice diagram

ones ((m-1)-dimensional cubes in the case of m variables with one non negated variable). We call the set of all marked points a covering.

Then we select the lowest 0-value points of the covering ((101)0) and we create (the disjunction of) the conjunction(s) of non negated variables ($x_1 \wedge x_3$) and negate it ($\neg(x_1 \wedge x_3)$). We mark all points that are greater than the selected ((111)0, cf. italics). Moreover, the marked points but the starting will be removed from the diagram. This is the β-reduction. Finally we build a conjunction from the obtained formulae ($(x_2 \vee x_3) \wedge \neg(x_1 \wedge x_3)$) that corresponds to the actual iteration step.

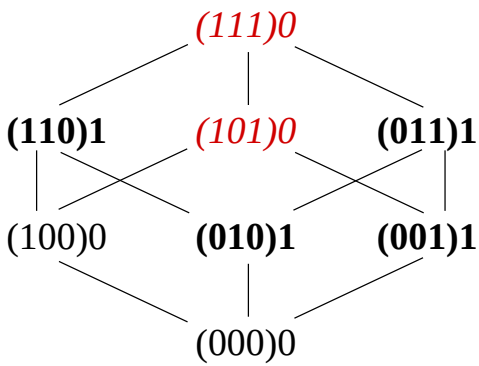

Figure 3. β-reduction of lattice diagram

We try to continue the iteration by starting with lowest 1-value points of the obtained diagram till no more changes can be carried out. (In our example there is no more iteration step.) Every formula resulted in an iteration step should

be connected by the previous one with disjunction. Finally we keep only the starting points of the last α-expansion and β-reduction.

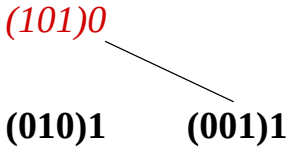

Figure 4. Result of the iteration process

Our operation can be fully reconstructed (characterized) by the remaining points of the diagram (see in [8]). The number of steps of the algorithm of reconstruction could serve as complexity.

Let be given an n-valued ($n > 2$) m-ary operation. It can be given by the lattice diagram of E^m, where E is the set of values. In this lattice diagram we take into account the natural ordering of the values together with the ordering according to the Hamming distance. The extension of the ICF algorithm to n-valued case is applied first to the lattice diagram, without constructing the formula describing the mapping. This is reasoned by the fact that the set of points greater than a given one cannot be described always by elementary conjunction. Moreover, the negation of a formula should have to mean the exclusion of all points covered by the concerned formula from the full space. (This is the problem in [FJ] respectively.) The simplification here can be carried out between n m-dimensional points where $m-1$ coordinates coincide and the remaining one takes all possible values (m is the number of variables).

In the n-valued case the iteration step means the following: First we select all the least points with value q ($q \neq 0$) from the diagram.

- If one of the selected points has at least one zero coordinate, in the lattice diagram we mark all points being greater than the starting ones regarding the $m-j$ zero coordinates ($(m-j)$-dimensional cubes in case of j nonzero coordinates). We call the set of all marked points a covering.
- If there is not any zero coordinate in the selected points, then we mark each point greater than the selected ones. This is also a covering.

These coverings are added to the ICF graph. A covering is characterized by the value of its initial point. If the value is q, the covering is called q-covering. This step is referred as an α-expansion.

In a q-covering we select the least points with value different from q. Then we mark in these coverings the points of the $(m-j)$ dimensional cube (if there

are $m - j$ zero coordinates in the selected points and all point greater than the selected otherwise. The sets of marked points (coverings) except for its initial points will be removed from the graph. This is the β-reduction.

We continue the iteration by starting with the covering prepared for the β-reduction. We make the α-expansion. The selected points will be now the least points with value different to the value of the initial point of the removed covering. Then the β-reduction is executed till no more change can be carried out.

In the example shown by Figure 5 we have one point (01)2 in the first step. The α-expansion consists of 3 points (boldface). Since the covering contains only 2-valued points, the β-reduction makes no changes.

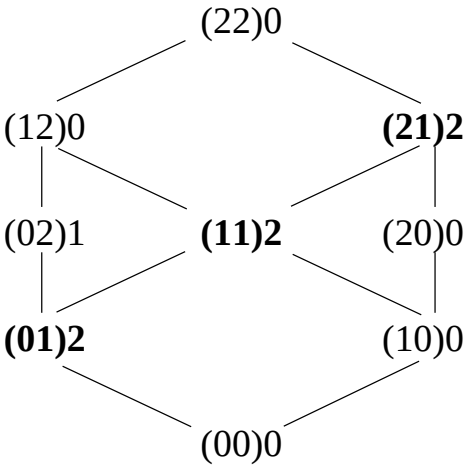

Figure 5. Lattice diagram of a three-valued operation

In the second step the selected point is (02)1. The α-expansion consists of 3 points (italics in Figure 6). The point (12)0 is the least among the covered non 1-valued points. However, its expansion can not assigned to a conjunction. It is only the set of all greater points including itself. These point will be removed from the diagram.

Since the removed points are 0-valued points and the iteration makes no more changes, the result diagram is in Figure 7. The operation can be fully reconstructed by the result diagram either. The number of steps of reconstruction of a formula from the diagram could serve as a measure for its complexity.

If the least nonzero-valued points in the original lattice diagram or in some part covered by an α-expansion have only nonzero coordinates (which can not occur in two-valued case) then these can not be the result of a simplification. However, it can happen that all the points greater than the selected belong to the same value. In this case, we keep the least point in the diagram (α-expansion).

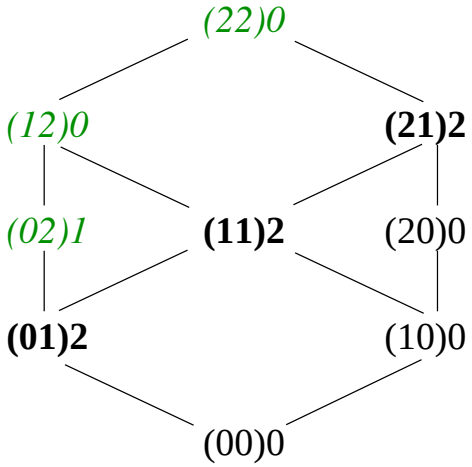

Figure 6. Second iteration step

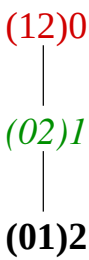

Figure 7. Result diagram

Performing the β-reduction we select the least nonzero-valued point and continue until a new iteration step impossible. The next example in Figures 8 and 9 shows such a case.

In the last example shown by Figure 10 we have two points $(02)1, (11)2$ in the first step.

The point (02)1 has a zero coordinate, so we mark the greater points (12)0 and (22)0. The point (11)2 has not any zero coordinate, so we mark the greater points ((21)2, (12)0 and (22)0. The α-expansion consists of 5 points. We select the least point ((12)0) with value different to 1 in the 1-covering and mark the greater one (22)0 (Figure 11).

The sets of marked points except its initial points will be removed from the graph (Figure 12).

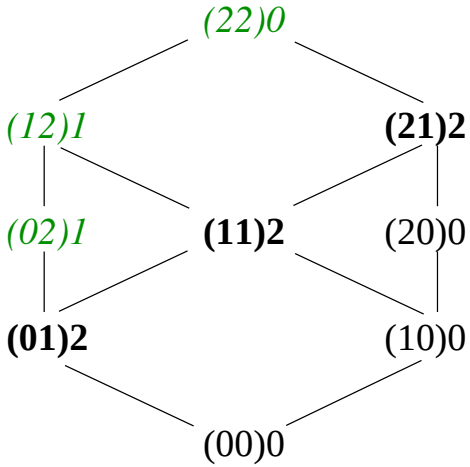

Figure 8. Iteration step in the second diagram

(22)0

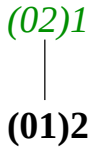

Figure 9. Second result diagram

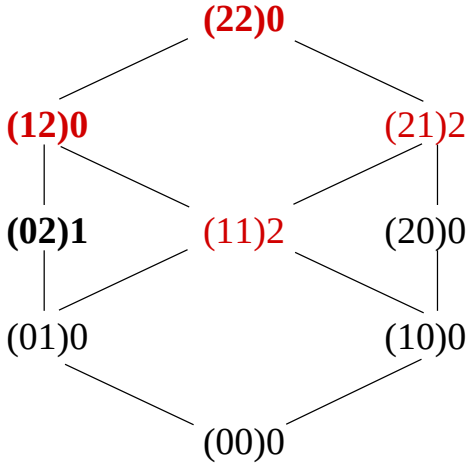

Figure 10. α-expansion in the third example

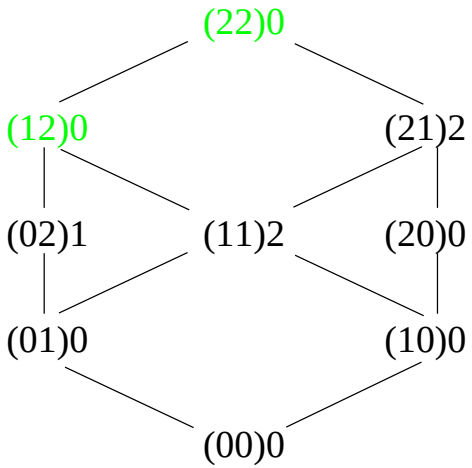

Figure 11. Before β-reduction

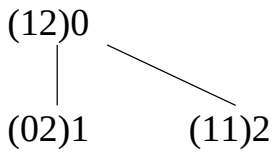

Figure 12. Third result diagram

Remember that the $\wedge$ and the $\vee$ operations correspond to the minimum and maximum operations in our n-valued logic. We note that negation has more extensions, however, there is no suitable negation to the description of the complementer of points covered by a formula with respect to the domain by a formula.

If we describe a phenomenon (a model) by a function of two- or many-valued logics then it looks reasonable to characterize changes of the phenomenon by a distance measure of the functions describing them. To the examination of the distance of two functions (n-valued operations), the comparison of the ICF diagrams can be more effective than that of the original lattice diagram.

7. Related work

In this paper we recalled the main properties of the ICF graph introduced originally in [10] and, proposed an ICF graph construction algorithm for many valued logical operations. Our construction was based on a set theoretic approach. An algorithm for the two valued case based on similar principles can

be found in [9], which doesn't attend by a description formula construction even in the two valued case. As for the many valued case, an algorithm using list management method and providing description formula can be seen in [4]. This looks well from formal point of view, however, the application of the included (mirror) negation operation for formulae is dubious.

In the present paper we showed that using elementary conjunction the point set obtained by the simplification of certain variables can be interpreted as an $n-j$ dimensional cube, while the complete conjunction expressed in terms of the threshold operation is an n-dimensional vector describing the set of points containing all points greater than the concerned one. Accordingly, the description formula for certain ICF graph can be simply determined during the construction of the graph. On the other hand, the description formula can be constructed similarly to 2-valued logic, the minimum, threshold, j_i and $\neg_k$ operations. For example the describing formula corresponding to Figure 4 is $(y \vee z) \wedge \neg(x \wedge z)$, while the formula to Figure 12 is $((1 \wedge j_2 y) \vee (2 \wedge x^1 \wedge y^1)) \wedge \neg_1 (0 \wedge x^1 \wedge y^2)$. Both the recently given and the previously known algorithms for the construction of the ICF graph together with the concerning theoretical background are of principal significance. Their efficiency and performance have not been investigated yet.

8. Summary

The obtained results mean a progress in the use of network diagrams at the creation of an ICF graph. This is important because the classical normal forms play a considerable role in the automated theorem proving. In the case of classical logics, booth the classification of the formulae into types α and β by Smullyan [16] and the generalized normal forms initiated by Fitting [3] meant a noteworthy progression. As it may be known from our papers [13, 14], so called point-plus and recursive rewriting systems can be introduced for the explicit construction of normal forms (c.f. Table 2). Nevertheless, there are some results for the synthesis of the tableaux method and the resolution calculus [5] as well.

These rewriting rules can be given even in many-valued logics if we apply the implication and the negation defined by Rossel in [15]. The main problem here arises at the definition of a complement pair. Namely, the conjunction of the complement pairs is some of the non-designated truth values instead of 0. Similarly, the resolution rule requires that the resolvent should be the consequence of the ancestors. This is the reason that the Gentzen sequent method is recently the highest developed one in the many-valued logic. The generalization of recursive rewriting rules for many valued logics could make it possible to treat the tableau and resolution calculi as dual calculi. In particular, [7] deals with the extension of

	$\neg\neg A \longmapsto A$	$\neg\top \longmapsto \bot \quad \neg\bot \longmapsto \top$
gC-rules:	$\langle[\alpha]\rangle \longmapsto \langle[\alpha_1],[\alpha_2]\rangle$	$\langle[\beta]\rangle \longmapsto \langle[\beta_1,\beta_2]\rangle$
gD-rules:	$[\langle\alpha\rangle] \longmapsto [\langle\alpha_1,\alpha_2\rangle]$	$[\langle\beta\rangle] \longmapsto [\langle\beta_1\rangle,\langle\beta_2\rangle]$

Table 2. Recursive rewriting rules

tableaux, sequent and resolution calculi, however, there are still a lot of problems in this field as well.

References

[1] **Bolc, L., Borowik, P.,** *Many-valued logics 1 (Theoretical foundation)*, Springer Verlag, 1992.

[2] **Bolc, L., Borowik, P.,** *Many-valued logics 2 (Automated reasoning and practical applications)*, Springer Verlag, 2003.

[3] **Fitting, M.,** *First-order logic and automated theorem proving*, Springer Verlag, 1996.

[4] **Frolov, A., Jakó, É.,** Algorithms for recognition of partially ordered objects and their application, *Scripta Technica*, **29**(1991), 109-118.

[5] **Kovásznai, G.,** HyperS tableaux - Heuristic hyper tableaux, *Acta Cybernetica*, **17**(2005), 325-338.

[6] **Hähnle, R.,** Short conjunctive normal forms in finitely-valued logics, *Journal of Logic and Computation*, 1994.

[7] **Hähnle, R., Escalada-Imaz, G.,** *Deduction in many-valued logics: a survey*, 1997.

[8] **Ittzés, P.,** *A discrete mathematical method and its application in biology*, PhD Thesis. ELTE TTK. 2006.

[9] **Jakó, É., Ittzés, P.,** A discrete mathematical approach to the analysis of spatial pattern. *Abstracta Botanica* **22**(1998), 121-142.

[10] **Jakó, É.,** *Iterative canonical decomposition of boolean functions and its application to logical design*, PhD Thesis. Technical University Moscow, 1983.

[11] **Panti, G.,** Multiple-valued logics, Chapter 2 of Volume 1 *Quantified Representation of Uncertainty and Imprecision* of a *Handbook of Defensible Reasoning and Uncertainty Management Systems*, Kluwer, 1995.

[12] **S. McCall** (ed.,) *Polish logic 1920-1939*, Oxford at the Calendron Press, 1967.

[13] **Pásztor Varga, K., Várterész, M.,** A generalized approach to the theorem proving methods. *Proc. 5th International Conference on Applied Informatics, Eger, 2001.* 191-200.

[14] **Pásztor Varga, K., Várterész, M.,** Comparison and usability of two rewriting systems for theorem proving. *Pure Mathematics and Applications* **13**(1-2)(2002), 293-302.

[15] **Rosser, J.B., Turquette, A.R.,** *Many-valued logics*, North Holland. 1952.

[16] **Smullyan, R.M.,** *First-order logic*, Springer-Verlag, 1968.

K. Pásztor Varga
Department of Programming Languages and Compilers,
Eötvös Loránd University
pkata@ludens.elte.hu

M. Várterész
Department of Computer Science,
University of Debrecen
varteres@inf.unideb.hu