Svoboda maps in many-valued logic

Milan Petrík

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
petrikm@cmp.felk.cvut.cz

Abstract

We present a method of finding optimized normal forms of functions in many-valued logic. Our approach is based on an extension of the set of logical connectives and a generalization of the technique of Svoboda maps. This tool may be directly applied to the design of many-valued logical circuits.

Keywords: Fuzzy logic, many-valued logic, sufficient set of logical connectives, fuzzy hardware, Svoboda map, disjunctive normal form.

MSC: Primary 03G10. Secondary 03B32, 03B70, 06D35, 03G10, 03G20.

1 Introduction

As an alternative to the classical two-valued logic, many-valued or fuzzy logic has been suggested as a tool which allows to express better the way of human reasoning and deduction [2, 3, 4]. Besides highly nontrivial mathematical results, it brought quite successful applications, especially in fuzzy control. Therefore the development of special hardware for faster implementations of fuzzy logic became topical. One of possible approaches is the use of many-valued logical circuits. Despite of many possibilities of its realization, one inevitably encounters a problem of design of fuzzy logical circuits from their smallest parts, logical gates.

In the design of classical two-valued circuits, gates performing basic logical operations are implemented and combined in some normal form (disjunctive or conjunctive) corresponding to the desired logical function. In order to find an optimally simplified expression, Carneaux or Svoboda maps are used as a standard tool.

*Work supported by the Czech Ministry of Education under Research Programme MSM 212300013 “Decision Making and Control in Manufacturing”, grant 201/02/1540 of the Grant Agency of the Czech Republic, and CEEPUS net SK-042.
Here we deal with the generalization to the case of many-valued logic, admitting more than two (but finitely many) logical values. The classical methods do not admit a canonical extension because the usual set of connectives (conjunction, disjunction, negation) is not sufficient. (See [1, 5] for a detailed analysis of possibilities of various fuzzy logical operations.) Therefore we extend the set of logical connectives. Then we show that we obtain a sufficient set of connectives which admits (disjunctive or conjunctive) normal forms. Moreover, it is possible to use Svoboda maps in order to obtain a simplified expression.

2 Basic definitions

Let us first introduce the basic notions used in the sequel.

We assume that the set of truth values is a finite totally ordered set, \( \mathcal{P} \). As only the order of truth values is important, without any loss of generality we shall represent them by rational numbers as follows: \( \mathcal{P} = \{0, \frac{1}{n-1}, \frac{2}{n-1}, \ldots, \frac{n-2}{n-1}, 1\} \). A logical variable may attain values from \( \mathcal{P} \). A logical function is a mapping \( f: \mathcal{P}^m \rightarrow \mathcal{P} \). The set of truth values has \( n \) elements and we speak of an \( n \)-valued logic, \( n \)-valued logical variable, etc.

We shall use the following operations on \( \mathcal{P} \) or its subsets:

Maximum: \( \lor: \mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P} \)
\[ a \lor b = \max(a, b) \]

Minimum: \( \land: \mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P} \)
\[ a \land b = \min(a, b) \]

Logical AND (conjunction): \( \cdot: \{0, 1\} \times \{0, 1\} \rightarrow \{0, 1\} \)
\[ a \cdot b = \begin{cases} 1 & a = 1 \text{ and } b = 1, \\ 0 & \text{otherwise}. \end{cases} \]

Connector: \( \ast: \mathcal{P} \times \{0, 1\} \rightarrow \mathcal{P} \)
\[ a \ast b = \begin{cases} a & \text{if } b = 1, \\ 0 & \text{if } b = 0. \end{cases} \]

Kronecker delta: \( \delta: \mathcal{P} \times \mathcal{P} \rightarrow \{0, 1\} \)
\[ \delta(a, b) = [a, b] = \begin{cases} 1 & a = b, \\ 0 & a \neq b. \end{cases} \]

Remark The connector and logical AND can be both replaced with the operation of minimum. In the sequel, we shall use the notation \([a, b]\) for Kronecker delta.

The following basic properties of these operations will be important in the sequel.
Proposition 2.1 Let $a, b, c \in \{0, 1\}$, $\vartheta \in \mathcal{P}$. Then:

1. $(a \cdot b) \lor (a \cdot c) = a \cdot (b \lor c)$ (distributivity)
2. $(a \lor b) \lor c = a \lor (b \lor c)$ (associativity)
3. $(\vartheta \cdot a) \lor (\vartheta \cdot b) = \vartheta \cdot (a \lor b)$ (distributivity)

Proof

1., 2. Maximum restricted to $\{0, 1\}$ coincides with the Boolean disjunction.
3. If at least one logical variable on the left-hand side is nonzero, the expression attains the value $\vartheta$, otherwise, it is 0. The same holds for the right-hand side.

Proposition 2.2 (properties of Kronecker delta) Let $\alpha, \beta \in \mathcal{P}$. Each logical variable $a$ satisfies:

4. $[a, \alpha] \cdot [a, \beta] = 0$ if $\alpha \neq \beta$
5. $[a, \alpha] \cdot [a, \beta] = [a, \alpha] = [a, \beta]$ if $\alpha = \beta$
6. $[a, \alpha] \lor [a, \beta] = [a, \alpha] = [a, \beta]$ if $\alpha = \beta$

Each logical variable $c$ attaining values in $\mathcal{P} = \{\gamma_1, \gamma_2, \ldots, \gamma_m\}$ satisfies

7. $[c, \gamma_1] \lor [c, \gamma_2] \lor \ldots \lor [c, \gamma_m] = 1$

Proof

4. At least one term of the conjunction attains 0, hence the whole expression vanishes.
5. As $\alpha = \beta$, both Kronecker delta functions are equal to each other and to the whole expression (which can hence be replaced by one of the Kronecker delta functions).
6. We apply the same argument as in 5.
7. If the set $\{\gamma_1, \gamma_2, \ldots, \gamma_m\}$ contains all values which $c$ may attain, then at least one of the Kronecker delta functions in the expression $[c, \gamma_1] \lor [c, \gamma_2] \lor \ldots \lor [c, \gamma_m]$ attains 1, as well as the whole expression.

The basic building block of the normal form is a P-term defined as follows:

Definition 2.3 (P-term and minterm) A P-term is an expression

$$\vartheta \ast ([a_1, \alpha_1] \cdot [a_2, \alpha_2] \cdot \ldots \cdot [a_m, \alpha_m])$$

where $a_1, a_2, \ldots, a_m$ are logical variables, $\alpha_1, \alpha_2, \ldots, \alpha_m \in \mathcal{P}$ are constants, and $\vartheta \in \mathcal{P}$ is a constant called the coefficient of P-term. The expression $[a_1, \alpha_1] \cdot [a_2, \alpha_2] \cdot \ldots \cdot [a_m, \alpha_m]$ is called the body of the P-term.

A minterm is a P-term that contains all logical variables.

Proposition 2.4 The only possible evaluations of a P-term are $\vartheta$ (iff $a_1 = \alpha_1, a_2 = \alpha_2, \ldots, a_m = \alpha_m$) and 0 (otherwise).

Proof If all equalities in the Kronecker deltas are satisfied, the body of the P-term is evaluated by 1 and the whole P-term equals $\vartheta$. If any of the equalities is violated, the body of the P-term, as well as the whole P-term, becomes 0.
3 Disjunctive normal forms

Let us first assume the classical normal forms composed from the Boolean con-nectives $\land, \lor, \neg, 0, 1$. These have an analogue in many-valued logic: the conjunction may be interpreted by the minimum, the disjunction by the maximum, and the negation by the function $x \mapsto 1 - x$. Nevertheless, these operations do not form a sufficient set of logical connectives. Indeed, the evaluation of a formula becomes always equal to the evaluation of some of the variables or its negation. This does not admit to represent all possible logical functions, even when we restrict to unary ones. One may think of a different interpretation of these connectives (using a triangular norm different from the minimum and a triangular conorm different from the maximum), but this does not solve the problem. Although there are operations admitting a large class of logical functions, still we do not obtain a sufficient set of logical connectives.

In our approach, the set of logical connectives is $\{*, \lor, \cdot, \delta\}$. The following theorem claims that our set of logical connectives is indeed sufficient and it gives us a normal form of a logical function.

**Theorem 3.1 (generalization of disjunctive normal form)** Every $n$-valued logical function, $f$, may be described by a table as follows:

<table>
<thead>
<tr>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$\ldots$</th>
<th>$a_m$</th>
<th>$f(a_1, a_2, \ldots, a_m)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha_{1,1}$</td>
<td>$\alpha_{1,2}$</td>
<td>$\ldots$</td>
<td>$\alpha_{1,m}$</td>
<td>$\vartheta_1$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\ddots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
</tr>
<tr>
<td>$\alpha_{t,1}$</td>
<td>$\alpha_{t,2}$</td>
<td>$\ldots$</td>
<td>$\alpha_{t,m}$</td>
<td>$\vartheta_t$</td>
</tr>
</tbody>
</table>

For the $i$-th row of the table, $i = 1, \ldots, l$, we construct a minterm

$$\mu_i = \vartheta_i \ast ([a_1, \alpha_{i,1}] \cdot [a_2, \alpha_{i,2}] \cdot \ldots \cdot [a_m, \alpha_{i,m}])$$

The expression

$$\mu_1 \lor \mu_2 \lor \ldots \lor \mu_l$$

has the same evaluation as $f(a_1, a_2, \ldots, a_m)$; we call it the disjunctive normal form (DNF).

**Proof** The $i$-th row of the table corresponds to the evaluation $a_1 = \alpha_{i,1}, a_2 = \alpha_{i,2}, \ldots, a_m = \alpha_{i,m}$, the corresponding minterm attains the value $\vartheta_i$, all other minterms are zero (Prop. 2.4). Taking the maximum of these arguments, the whole DNF attains the value $\vartheta_i$ in this case. This holds for any $i$, so the proof is complete.

**Corollary** In any $n$-valued logic, $n \in \mathbb{N}$, $n \geq 2$, the set $\{*, \lor, \cdot, \delta\}$ is a sufficient set of logical connectives.

Let us demonstrate the latter tool on an example.
Example 3.2 (construction of DNF from table) Let us consider three-valued logic, \( \mathcal{P} = \{0, \frac{1}{2}, 1\} \), and a three-valued logical function given by the following table. Find its DNF and simplify it.

<table>
<thead>
<tr>
<th>row no.</th>
<th>b</th>
<th>a</th>
<th>( f(a, b) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>( \frac{1}{2} )</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>( \frac{1}{2} )</td>
</tr>
<tr>
<td>4</td>
<td>( \frac{1}{2} )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>( \frac{1}{2} )</td>
<td>1</td>
<td>( \frac{1}{2} )</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>( \frac{1}{2} )</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

Solution

For each row of the table, we construct the corresponding minterm:

<table>
<thead>
<tr>
<th>row no.</th>
<th>b</th>
<th>a</th>
<th>( f(a, b) )</th>
<th>minterm</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>( (b, 0) \cdot [a, 0] )</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>( \frac{1}{2} )</td>
<td>1</td>
<td>( (b, 0) \cdot [a, \frac{1}{2}] )</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>( \frac{1}{2} )</td>
<td>( (b, 0) \cdot [a, 1] )</td>
</tr>
<tr>
<td>4</td>
<td>( \frac{1}{2} )</td>
<td>0</td>
<td>0</td>
<td>( (b, \frac{1}{2}) \cdot [a, 0] )</td>
</tr>
<tr>
<td>5</td>
<td>( \frac{1}{2} )</td>
<td>( \frac{1}{2} )</td>
<td>0</td>
<td>( (b, \frac{1}{2}) \cdot [a, \frac{1}{2}] )</td>
</tr>
<tr>
<td>6</td>
<td>( \frac{1}{2} )</td>
<td>1</td>
<td>( \frac{1}{2} )</td>
<td>( (b, \frac{1}{2}) \cdot [a, 1] )</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>( (b, 1) \cdot [a, 0] )</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>( \frac{1}{2} )</td>
<td>0</td>
<td>( (b, 1) \cdot [a, \frac{1}{2}] )</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>( \frac{1}{2} )</td>
<td>( (b, 1) \cdot [a, 1] )</td>
</tr>
</tbody>
</table>

Minterms with coefficient 0 are always equal to 0 and may be omitted; then the DNF achieves the following form:

\[
 f(a, b) = \frac{1}{2} \cdot (b, 0) \cdot [a, 1] \vee \frac{1}{2} \cdot (b, \frac{1}{2}) \cdot [a, \frac{1}{2}] 
\vee \frac{1}{2} \cdot (b, \frac{1}{2}) \cdot [a, 1] \vee 1 \cdot (b, 0) \cdot [a, \frac{1}{2}] 
\vee 1 \cdot (b, 1) \cdot [a, 0]
\]

We simplify this expression using distributivity:

\[
 f(a, b) = \frac{1}{2} \cdot \left( (b, 0) \cdot [a, 1] \vee (b, \frac{1}{2}) \cdot [a, 1] \vee (b, 1) \cdot [a, 1] \right) 
\vee 1 \cdot (b, 0) \cdot [a, \frac{1}{2}] 
\vee 1 \cdot (b, 1) \cdot [a, 0]
\]

\[
 f(a, b) = \frac{1}{2} \cdot \left( [a, 1] \cdot \left( [b, 0] \vee [b, \frac{1}{2}] \vee [b, 1] \right) \right) 
\vee 1 \cdot (b, 0) \cdot [a, \frac{1}{2}] 
\vee 1 \cdot (b, 1) \cdot [a, 0]
\]

5
According to Theorem 2.2, the expression \((\lceil b, 0 \rceil \lor \lceil b, \frac{1}{2} \rceil \lor \lceil b, 1 \rceil)\) equals 1. Thus the whole expression may be simplified to

\[
f(a, b) = \frac{1}{2} \ast (\lceil [a, 1] \rceil) \lor 1 \ast (\lceil b, 0 \rceil \cdot \lceil a, \frac{1}{2} \rceil) \lor 1 \ast (\lceil b, 1 \rceil \cdot \lceil a, 0 \rceil)
\]

**Proposition 3.3** In any \(n\)-valued logic, the set \(\{\land, \lor, \delta\}\) is a sufficient set of logical connectives.

**Proof** Operation \(\land\) represents a common generalization of \(\ast\) and \(\cdot\).

**Definition 3.4** We introduce the following logical operations:

**Negation:** \(\neg: \mathcal{P} \to \mathcal{P}\)

\[\neg a = 1 - a\]

**Shaffer operator (NAND):** \(\uparrow: \mathcal{P} \times \mathcal{P} \to \mathcal{P}\)

\[a \uparrow b = -(a \land b) = 1 - \min(a, b)\]

**Implication:** \(\Rightarrow: \mathcal{P} \times \mathcal{P} \to \mathcal{P}\)

\[a \Rightarrow b = (-a) \lor b = \max(1 - a, b)\]

**Proposition 3.5** (Properties of the Shaffer operator) Let \(a, b \in \mathcal{P}\). Then:

8. \(a \uparrow a = \neg a\)
9. \(a \uparrow 0 = 1\)
10. \(a \uparrow 1 = a \uparrow a\)
11. \((a \uparrow a) \uparrow (a \uparrow a) = a\)
12. \(a \uparrow b = b \uparrow a\)

**Proof**

8. \(1 - \min(a, a) = 1 - a = \neg a\)
9. \(1 - \min(a, 0) = 1 - 0 = 1\)
10. \(1 - \min(a, 1) = 1 - a = \neg a = a \uparrow a\)
11. \((a \uparrow a) \uparrow (a \uparrow a) = (-a) \uparrow (-a) = \neg(a) = a\)
12. The operation \(\land\) is commutative and so is the operation \(\uparrow\).

**Proposition 3.6** In any \(n\)-valued logic, \(n \in \mathcal{N}, n \geq 2\), the set \(\{\uparrow, \delta\}\) is a sufficient set of logical connectives.

**Proof** Operations \(\land\) and \(\lor\) can be represented by \(\uparrow\):

\[
\neg a = a \uparrow a
\]

\[a \land b = -(a \uparrow b)\]

\[a \lor b = -(\neg a \land \neg b)\]
Proposition 3.7 In any \( n \)-valued logic, \( n \in \mathbb{N} \), \( n \geq 2 \), the set \( \{ \Rightarrow, 0, \delta \} \) is a sufficient set of logical connectives.

Proof Operation \( \uparrow \) can be represented by \( \Rightarrow \) and 0:
\[
\begin{align*}
\neg a &= a \Rightarrow 0 \\
(a \uparrow b) &= a \Rightarrow (\neg b)
\end{align*}
\]

4 Maps

The generalisation of Carnaugh maps is not possible, because they use the Gray code that doesn’t exist in many-valued logic. We have to use Svoboda maps [6] (also called Veitch or Marquand maps) that use the incremental code. In this section, we introduce a generalization of these maps to many-valued logic and show their use for simplification of disjunctive normal forms.

Definition 4.1 Let \( P \) be the set of truth values of an \( n \)-valued logic and let \( f: P^m \rightarrow P \) be an \( n \)-valued logical function.

The map of \( f \) is a table with \( n^m \) entries; each entry represents the evaluation of the function \( f(a_1, \ldots, a_m) \) for one of \( n^m \) possible evaluations of \( m \) logical variables \( a_1, \ldots, a_m \).

In particular, the Svoboda map of function \( f \) is obtained as follows: We divide the variables into two disjoint sets, one for row indices, the other for column indices. Rows and columns are indexed by all possible evaluations of the respective variables, ordered lexicographically.

A set of entries of the map of \( f \) is called a loop iff it satisfies the following conditions:

1. all values of the entries of the loop are equal,
2. the loop forms a (discrete) \( k \)-dimensional cube, \( k \in \mathbb{N} \), with edge of length \( n \).

If we relax the first condition, we speak of a mixed loop.

The value of the loop is the value of its entries.

The number \( k \) is called the dimension of the loop.

A set of loops is called a covering iff each entry belongs to at least one loop.

A minimal covering is a covering such that none of its proper subsets is a covering.

Theorem 4.2 (construction of DNF using map) Let \( f: P^m \rightarrow P \) be an \( n \)-valued logical function. To each covering of a map of \( f \) we associate an expression as follows: The \( i \)-th loop of the covering is associated with the \( P \)-term
\[
\mu_i = \partial_i \ast \left( [a_{i,1}, \alpha_{i,1}] \cdot [a_{i,2}, \alpha_{i,2}] \cdot \ldots \cdot [a_{i,l_i}, \alpha_{i,l_i}] \right), \quad i = 1 \ldots s,
\]
where
• $s$ is the number of loops in the covering,

• $a_{i,1}, a_{i,2}, \ldots, a_{i,l_i}$ are those variables which are constant over all entries of the $i$-th loop (attaining values $\alpha_{i,1}, \alpha_{i,2}, \ldots, \alpha_{i,l_i}$, respectively),

• $\vartheta_i$ is the value of the loop,

• $l_i = m - k_i$,

• $k_i$ is the dimension of the $i$-th loop.

The expression

$$\mu_1 \vee \mu_2 \vee \ldots \vee \mu_l$$

has the same evaluation as $f(a_1, a_2, \ldots, a_m)$.

**Proof** Each evaluation of the input variables, $a_1 = \alpha_1, a_2 = \alpha_2, \ldots, a_s = \alpha_s$ corresponds to an entry with value $f(\alpha_1, \alpha_2, \ldots, \alpha_s) = \beta$. As we started from a covering, each entry is contained in a loop. Each loop containing this entry has the same coefficient $\beta$. This is also the evaluation of all P-terms covering this entry. All P-terms not covering this entry are evaluated to zero. Taking the maximum over all P-terms gives $\beta$, thus $f(\alpha_1, \alpha_2, \ldots, \alpha_s) = \beta$. □

**Example 4.3 (construction of DNF using map)** A logical function is given by Svoboda map. Find its DNF and simplify it.

**Solution** We find a minimal covering:
The P-terms corresponding to the loops are:

1st loop: \( \frac{1}{2} \cdot [b, 1] \cdot [d, 1] \)

2nd loop: \( \frac{1}{2} \cdot [a, 1] \cdot [c, \frac{1}{2}] \)

3rd loop: \( 1 \cdot [a, \frac{1}{2}] \cdot [b, \frac{1}{2}] \cdot [c, 1] \)

4th loop: \( 1 \cdot [a, 0] \cdot [b, 0] \cdot [d, \frac{1}{2}] \)

5th loop: \( 1 \cdot [b, 0] \cdot [c, 1] \cdot [d, \frac{1}{2}] \)

6th loop: \( 1 \cdot [a, 0] \cdot [b, 0] \cdot [c, 1] \cdot [d, 1] \)

The simplified DNF becomes

\[
f(a, b, c, d) = \frac{1}{2} \cdot [b, 1] \cdot [d, 1] \lor \frac{1}{2} \cdot [a, 1] \cdot [c, \frac{1}{2}] \lor 1 \cdot [a, \frac{1}{2}] \cdot [b, \frac{1}{2}] \cdot [c, 1] \lor 1 \cdot [a, 0] \cdot [b, 0] \cdot [d, \frac{1}{2}] \lor 1 \cdot [b, 0] \cdot [c, 1] \cdot [d, \frac{1}{2}] \lor 1 \cdot [a, 0] \cdot [b, 0] \cdot [c, 1] \cdot [d, 1]
\]

In order to obtain a simple expression, we need a covering with a small number of large loops. If we have many logical values, the large loops occur quite rarely. Therefore it is desirable to generalize our approach to mixed loops, too. We shall do that in the rest of this section. We start with generalizations of the respective definitions.

**Definition 4.4 (value of mixed loop)** A value of the mixed loop is the smallest value in the loop.
Definition 4.5 (mixed covering) A set of mixed loops is called a mixed covering iff

- each entry belongs to at least one mixed loop,
- the value of $f$ at each entry is the maximum of values of all mixed loops containing this entry.

A minimal mixed covering is a mixed covering such that none of its proper subsets is a mixed covering.

Theorem 4.6 (construction of DNF using map with mixed loops)

Let $f: P^m \rightarrow P$ be an $n$-valued logical function. To each mixed covering of a map of $f$ we associate an expression as follows: The $i$-th mixed loop of the covering is associated with the $P$-term

$$
\mu_i = \vartheta_i \ast \left( [a_{i,1}, \alpha_{i,1}] \cdot [a_{i,2}, \alpha_{i,2}] \cdot \ldots \cdot [a_{i,l_i}, \alpha_{i,l_i}] \right), \quad i = 1 \ldots s,
$$

where

- $s$ is the number of loops in the covering,
- $a_{i,1}, a_{i,2}, \ldots, a_{i,l_i}$ are those variables which are constant over all entries of the $i$-th loop (attaining values $\alpha_{i,1}, \alpha_{i,2}, \ldots, \alpha_{i,l_i}$, respectively),
- $\vartheta_i$ is the value of the loop,
- $l_i = m - k_i$,
- $k_i$ is the dimension of the $i$-th loop.

The expression

$$
\mu_1 \lor \mu_2 \lor \ldots \lor \mu_t
$$

has the same evaluation as $f(a_1,a_2,\ldots,a_m)$.

Proof Each evaluation of the input variables, $a_1 = \alpha_1, a_2 = \alpha_2, \ldots, a_s = \alpha_s$ corresponds to an entry with value $f(\alpha_1, \alpha_2, \ldots, \alpha_s) = \beta$. As we started from a mixed covering, each entry is contained in a mixed loop. The maximal evaluation of all P-terms covering this entry equals $\beta$. All P-terms not covering this entry are evaluated to zero, thus $f(\alpha_1, \alpha_2, \ldots, \alpha_s) = \beta$.

Example 4.7 (construction of DNF using mixed loops) A logical function is given by Svoboda map. Find its DNF and simplify it.
We take a mixed loop which covers values $\frac{1}{2}$ and also one value 1. The corresponding P-term is

$$\frac{1}{2} \cdot \left( [b, \frac{1}{2}] \right)$$

Following the second condition, the unit covered by the first loop has to be covered also by another loop of value 1.
The corresponding DNF becomes

\[ f(a, b, c) = \frac{3}{4} \cdot \left( [b, \frac{3}{4}] \right) \lor 1 \cdot \left( [a, 1] \cdot [c, \frac{3}{4}] \right) \]

5 Conclusion

We suggested a sufficient set of logical connectives for a many-valued logic. We proved that they admit a generalization of the technique using Svoboda maps. Examples are presented which show that this tool can be easily applied to finding a simplified disjunctive normal form. Our approach may be directly applied to the design of many-valued logical circuits. We did not discuss a concrete possible hardware implementation of many-valued logic, but applicability of our results is independent of it.

References


Milan Petrík, born in Prague, student of Computer Science at the Czech Technical University in Prague, Faculty of Electrical Engineering, supervisor Associated Professor Mirko Navara; now at a fellowship at the University of Patras, Greece. Interested in fuzzy logic and hardware, theoretical computer science and theory of compilers.