On the Complexity of Classification Functions

Tsutomu Sasao
Department of Computer Science and Electronics,
Kyushu Institute of Technology,
Iizuka 820-8502, Japan

Abstract

A classification function is a multiple-valued input function specified by a set of rules, where each rule is a conjunction of range functions. The function is useful for packet classification for internet, network intrusion detection system, etc. This paper considers the complexity of range functions and classification functions represented by sum-of-products expressions of binary variables. It gives tighter upper bounds on the number of products for range functions.

1. Introduction

A classification function is used for packet classification in the internet [3], where internet service providers (ISPs) want to provide differentiated services to various users. Classification functions are also used for network intrusion detection system (IDS). Since high-speed processing is necessary, various hardware implementations have been proposed [1, 6, 7, 12, 14].

In this paper, we derive an upper bound on the number of products in sum-of-products expressions (SOPs) to represent classification functions. With this bound, we can estimate the size of a circuit for the packet classification. As for the hardware to implement the classification functions, we assume an off-the-shelf Ternary Content Addressable Memory (TCAM) [7], and standard memory. The number of products in an SOP gives the number of words in the TCAM.

A classification function is defined by a set of rules, and each rule is a conjunction of range functions. For example, consider the classification function

$$f : \{0, 1, 2, 3\} \times \{0, 1, 2, 3\} \rightarrow \{0, 1, 2\}$$

consisting of two rules:

$$R_1 = (0 \leq X_1 \leq 2) \cdot (1 \leq X_2 \leq 3),$$

and

$$R_2 = (1 \leq X_1 \leq 3) \cdot (0 \leq X_2 \leq 2).$$

In this case, '$0 \leq X_i \leq 2$' and '$1 \leq X_i \leq 3$' are range functions. $f = 1$ if R_1 holds, $f = 2$ if R_1 does not hold, but R_2 holds, and $f = 0$ if neither R_1 nor R_2 holds.
In our method, we use an SOP of binary variables to represent a range function. The rules are stored in the TCAM array in the order of decreasing priority. Suppose that X_1 and X_2 are represented by (x_1, x_2), and (x_3, x_4), respectively. That is, $X_1 = 2x_1 + x_2$, and $X_2 = 2x_3 + x_4$, where $+$ denotes an integer addition. Then, the range function '$0 \leq X_1 \leq 2$' is represented by the SOP: $\bar{x}_1 \vee \bar{x}_2$. In a similar way, the range function '$1 \leq X_1 \leq 3$' is represented by the SOP: $x_1 \vee x_2$. In this example, rules have two *fields*, and each field of a rule is represented by an SOP with two products. For example, R_1 can be represented by $R_1 = (\bar{x}_1 \vee \bar{x}_2)(x_3 \vee x_4) = \bar{x}_1 x_3 \vee \bar{x}_1 x_4 \vee \bar{x}_2 x_3 \vee \bar{x}_2 x_4$. Thus, the number of products to represent R_1 by an SOP is $2 \times 2 = 4$.

In the real packet classification, the numbers of values of the variable are either 2^{32}, 2^{16} or 2^8, and the numbers of variables are 5 to 8. So, the size of the SOP can be very large. This is the reason why we are interested in the complexity of SOPs for range functions and classification functions. Simplification of a set of rules of the a classification function is related to the minimization of SOPs.

The direct method to represent the range [A,B] is to store the pair of integers. However, this method requires a comparator of values in each field [12], and conventional memory cannot be used. Another method is to use a Look-Up-Table (LUT) for each field [6]. However, this can be expensive when the number of bits in a field is large. In many cases, we have to update rules of the classification functions frequently. This is the reason why CAMs are often used in the network applications. This paper gives tight upper bounds on the numbers of products in SOPs for range functions. Note that a CAM word corresponds to a product in an SOP.

This paper is organized as follows: Section 2 defines range functions, and shows their properties. Section 3 considers the number of products to represent a range function by an SOP. Section 4 defines classification functions, and shows some examples. Section 5 considers the complexity of classification functions represented by SOPs. Section 6 shows an application of classification functions in the internet. And, finally Section 7 concludes the paper.

2. Range Functions

A range function is a generalization of a comparator function. To define the range function, we use a *Greater-than-or-Equal-to function* and a *Less-than-or-Equal-to function*.

Definition 2.1 *An n-input* **GE function** *(Greater-than-or-Equal-to function) is*

$$GE(n:A) = \begin{cases} 1 & \text{if } X \geq A \\ 0 & \text{otherwise} \end{cases}$$

where $X = \sum_{i=0}^{n-1} x_i \cdot 2^i$, $\vec{x} = (x_{n-1}, x_{n-2}, \ldots, x_1, x_0)$ is the binary input vector, X is an integer represented by $\vec{x}$, and A is an integer such that $0 \leq A \leq 2^n - 1$.

Definition 2.2 *An n-input* **LE function** *(Less-than-or-Equal-to function) is*

$$LE(n:B) = \begin{cases} 1 & \text{if } X \leq B \\ 0 & \text{otherwise} \end{cases}$$

where X is an integer represented by $\vec{x}$, and B is an integer such that $0 \leq B \leq 2^n - 1$.

Definition 2.3 *An n-input* **range function** *is*

$$RA(n:A,B) = \begin{cases} 1 & \text{if } A \leq X \leq B \\ 0 & \text{otherwise} \end{cases}$$

where X is an integer represented by $\vec{x}$, and A and B are integers such that $0 \leq A \leq B \leq 2^n - 1$.

Example 2.1 *Consider the case of $n = 4$, $A = 5$, and $B = 10$. Table 2.1 shows $GE(4:5)$, $LE(4:10)$, and $RA(4:5,10)$.* *(End of Example)*

From the definitions, we have the following:

Lemma 2.1

$$RA(n:A,B) = GE(n:A) \cdot LE(n:B).$$

3. Complexity of Range Functions

In this part, we consider sum-of-products expressions to represent range functions.

Table 2.1. Examples of GE, LE, and RA functions.

8 x_3	4 x_2	2 x_1	1 x_0	$GE(4:5)$	$LE(4:10)$	$RA(4:5,10)$
0	0	0	0	0	1	0
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	0	1	0
0	1	0	1	1	1	1
0	1	1	0	1	1	1
0	1	1	1	1	1	1
1	0	0	0	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	1	1
1	0	1	1	1	0	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	1	0	0

Definition 3.1 *Let x be a variable. Then, $\bar{x}$ is a complement of the variable. x and $\bar{x}$ are* **literals** *of a variable x. The AND of literals is a* **product**. *A* **minterm** *is logical product of n literals where each variable occurs as exactly one literal. The OR of products is a* **sum-of-products expression** *(***SOP***). Let two functions be f and g. f* **implies** *g if every $\vec{x}$ satisfying $f(\vec{x}) = 1$ satisfies also $g(\vec{x}) = 1$. A minterm that implies f is a* **minterm of** *f. A* **prime implicant** *(***PI***) of a function f is a product that implies f, such that the deletion of any literal from the product results in a new product that does not imply f. An* **irredundant sum-of-products expression** *(***ISOP***) is an SOP, where each product is a PI, and no PI can be deleted without changing the function represented by the expression. The* **size** *of an SOP is the number of PIs in the SOP. Among the ISOPs for f, the ISOP with the minimum size is a* **minimum SOP** *(***MSOP***). The size of a MSOP of function f is denoted as $\tau(f)$.*

For a given n, we are interested in the most complicated range function. That is, the function with the largest $\tau(RA(\vec{x}:A,B))$.

Definition 3.2 *$\mu(n)$ denotes the number of products in an MSOP for the n-variable range function with the largest number of products.*

By exhaustive examination, we have the following:

Theorem 3.1 $\mu(n) = n, \ (n = 1,2,3,4)$.

Example 3.1 *Most complicated range functions up to $n = 4$ include:*
$RA(1:1,1) = x_0$,
$RA(2:1,2) = x_1\bar{x}_0 \vee \bar{x}_1 x_0$,
$RA(3:1,7) = x_2 \vee x_1 \vee x_0$,
$RA(3:1,6) = \bar{x}_2 x_1 \vee \bar{x}_1 x_0 \vee \bar{x}_0 x_2$,
$RA(4:1,15) = x_3 \vee x_2 \vee x_1 \vee x_0$,
$RA(4:5,10) = \bar{x}_3 x_2 x_0 \vee \bar{x}_3 x_2 x_1 \vee x_3 \bar{x}_2 \bar{x}_0 \vee x_3 \bar{x}_2 \bar{x}_1$.
(End of Example)

One might conjecture that Theorem 3.1 is true for larger n, but it is not. From here, we are going to show that $\mu(n) = 2(n-2)$ for $n \geq 5$ (Theorem 3.2).

Lemma 3.1 *$GE(n : A)$ can be represented by an SOP with at most $1 + \sum_{i=1}^{n-1} \bar{a}_i$ disjoint products, where $\vec{a} = (a_{n-1}, a_{n-2}, \ldots, a_1, a_0)$ is the binary representation of A.*

(Proof) $GE(n : A)$ can be represented as:
$GE(n : A) = x_{n-1}\bar{a}_{n-1} \vee (x_{n-1} \equiv a_{n-1})x_{n-2}\bar{a}_{n-2} \vee (x_{n-1} \equiv a_{n-1})(x_{n-2} \equiv a_{n-2})x_{n-3}\bar{a}_{n-3} \vee (x_{n-1} \equiv a_{n-1})(x_{n-2} \equiv a_{n-2})(x_{n-3} \equiv a_{n-3})x_{n-4}\bar{a}_{n-4} \vee \cdots (x_{n-1} \equiv a_{n-1})(x_{n-2} \equiv a_{n-2})(x_{n-3} \equiv a_{n-3}) \cdots (x_1 \equiv a_1)(x_0 \vee \bar{a}_0)$, where $A = \sum_{i=0}^{n-1} a_i 2^i$. Also, the symbol $\equiv$ denotes the equivalence operator. Note that the number of products in the SOP is at most $1 + \sum_{i=1}^{n-1} \bar{a}_i$. Especially, $GE(n : 1)$ requires n products. (Q.E.D.)

Lemma 3.2 *$LE(n : B)$ can be represented by an SOP with at most $1 + \sum_{i=1}^{n-1} b_i$ disjoint products, where $\vec{b} = (b_{n-1}, b_{n-2}, \ldots, b_1, b_0)$ is the binary representation of B.*

(Proof) $LE(n : B)$ can be represented as
$LE(n, B) = \bar{x}_{n-1}b_{n-1} \vee (x_{n-1} \equiv b_{n-1})\bar{x}_{n-2}b_{n-2} \vee (x_{n-1} \equiv b_{n-1})(x_{n-2} \equiv b_{n-2})\bar{x}_{n-3}b_{n-3} \vee (x_{n-1} \equiv b_{n-1})(x_{n-2} \equiv b_{n-2})(x_{n-3} \equiv b_{n-3})\bar{x}_{n-4}b_{n-4} \vee \cdots (x_{n-1} \equiv b_{n-1})(x_{n-2} \equiv b_{n-2})(x_{n-3} \equiv b_{n-3}) \cdots (x_1 \equiv b_1)(\bar{x}_0 \vee b_0)$, where $B = \sum_{i=n-1}^{0} b_i 2^i$. Note that the number of products in the SOP is at most $1+\sum_{i=1}^{n-1} b_i$. Especially, $LE(n : 2^n - 2)$ requires n products. (Q.E.D.)

Lemma 3.3 $\bar{x}_1(x_2 \vee x_3 \vee \cdots \vee x_n) \vee x_1(\bar{x}_2 \vee \bar{x}_3 \vee \cdots \vee \bar{x}_n) = x_1\bar{x}_2 \vee x_2\bar{x}_3 \vee \cdots \vee x_n\bar{x}_1$, *where $n \geq 2$.*

(Proof) First, consider the complement of the left-hand-side of the equation: $(x_1 \vee \bar{x}_2\bar{x}_3 \ldots \bar{x}_n) \cdot (\bar{x}_1 \vee x_2 x_3 \cdots x_n) = \bar{x}_1\bar{x}_2\bar{x}_3 \cdots \bar{x}_n \vee x_1 x_2 x_3 \cdots x_n$
Thus, by De Morgan's Theorem, the left-hand-side of the equation is equal to
$(x_1 \vee x_2 \vee x_3 \vee \cdots \vee x_n) \cdot (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \cdots \vee \bar{x}_n)$.
Also, note that the function is symmetric. Let $x_1 = 0$, then the left-hand-side will be $x_2 \vee x_3 \vee \cdots \vee x_n$. Let $x_1 = 1$ then, the left-hand-side will be $\bar{x}_2 \vee \bar{x}_3 \vee \cdots \vee \bar{x}_n$. We can confirm that the same properties hold for the right-hand-side of the equation. Thus, we have the lemma. (Q.E.D.)

Lemma 3.4 *For $n \geq 4$, $RA(n : A, B)$ can be represented by an SOP with at most $2(n-2)$ products.*

(Proof) First, we show that the number of the products is at most $2(n-1)$. We use mathematical induction to prove this. When $n = 4$, the lemma holds. Assume that the lemma holds for k-variable range functions. That is, any range function with k variables can be represented by an SOP with at most $2(k-1)$ products. Next, consider the case of $k+1$ variables. Since $A \leq B$, we need only to consider the following three cases:
When $a_k = 0$ and $b_k = 0$,
$RA(k+1 : A, B) = \bar{x}_k GE(k : A')LE(k : B')$
$= \bar{x}_k RA(k : A', B')$, where $A' = A$ and $B' = B$.
When $a_k = 0$ and $b_k = 1$,
$RA(k+1 : A, B) = \bar{x}_k GE(k : A') \vee x_k LE(k : B')$,
where $A' = A$ and $B' = B - 2^k$.
When $a_k = 1$ and $b_k = 1$,
$RA(k+1 : A, B) = x_k GE(k : A')LE(k : B')$
$= x_k RA(k : A', B')$, where $A' = A - 2^k$ and $B' = B - 2^k$.
In the first and the third case, the number of products is at most $2(k-1)$, by the hypothesis of induction. In the second case, the number of products is at most $\sum_{i=1}^{k-1} \bar{a}_i + \sum_{i=1}^{k-1} b_i + 2 = \sum_{i=1}^{k-1} (\bar{a}_i + b_i) + 2$. Thus, $\tau(RA(k+1 : A, B)) \leq 2k$. Hence, the number of the products in an SOP for n-variable range function is at most $2(n-1)$.
Second, we show that the number of the products is at most $2(n-2)$.
Consider the case, where the number of the products can be $2(n-1)$. From the above observation, this case can happen only when
$(a_{n-1}, a_{n-2}, \ldots, a_1, a_0) = (0, 0, 0, \ldots, 0, 1)$, and
$(b_{n-1}, b_{n-2}, \ldots, b_1, b_0) = (1, 1, 1, \ldots, 1, 0)$.
In this case,
$RA(n : 1, 2^n - 2) = GE(n : 1) \cdot LE(n : 2^n - 2)$, where
$GE(n : 1) = x_{n-1} \vee \bar{x}_{n-1}(x_{n-2} \vee \bar{x}_{n-2}x_{n-3} \vee \cdots \vee \bar{x}_{n-2}\bar{x}_{n-3} \cdots \bar{x}_1 x_0) = x_{n-1} \vee \bar{x}_{n-1}(x_{n-2} \vee \cdots \vee x_0)$,
$LE(n : 2^n - 2) = \bar{x}_{n-1} \vee x_{n-1}(\bar{x}_{n-2} \vee x_{n-2}\bar{x}_{n-3} \vee \cdots \vee x_{n-2}x_{n-3} \cdots x_1 \bar{x}_0) = \bar{x}_{n-1} \vee x_{n-1}(\bar{x}_{n-2} \vee \cdots \vee \bar{x}_0)$,
$RA(n : 1, 2^n - 2) = \bar{x}_{n-1}(x_{n-2} \vee x_{n-3} \vee \cdots x_0) \vee x_{n-1}(\bar{x}_{n-2} \vee \bar{x}_{n-3} \vee \cdots \vee \bar{x}_0)$.
By Lemma 3.3, we have
$RA(n : 1, 2^n - 2) = x_{n-1}\bar{x}_{n-2} \vee x_{n-2}\bar{x}_{n-3} \vee \cdots \vee x_1\bar{x}_0 \vee x_0\bar{x}_{n-1}$.
Note that the function can be represented by only n products. Thus, there exist no case that requires $2(n-1)$ products.
Next, consider the case where the number of the products can be $2n-3$. From the above observation, this case can happen when
$(a_{n-1}, a_{n-2}, \ldots, a_1, a_0) = (0, 0, 0, \ldots, 0, \mathbf{1}, 0, \ldots, 0, 1)$,
$(b_{n-1}, b_{n-2}, \ldots, b_1, b_0) = (1, 1, 1, \ldots, 1, \mathbf{1}, 1, \ldots, 1, 0)$,
or
$(a_{n-1}, a_{n-2}, \ldots, a_1, a_0) = (0, 0, 0, \ldots, 0, \mathbf{0}, 0, \ldots, 0, 1)$,

$(b_{n-1}, b_{n-2}, \ldots, b_1, b_0) = (1, 1, 1, \ldots, 1, \mathbf{0}, 1, \ldots, 1, 0)$, where the t-th elements of the vectors are highlighted by boldface, and the LSB is the 0-th bit. Consider the first case:
$RA(n : 1+2^t, 2^n-2) = GE(n : 1+2^t) \cdot LE(n : 2^n-2)$, where
$GE(n : 1+2^t) = x_{n-1} \vee \bar{x}_{n-1}(x_{n-2} \vee \bar{x}_{n-2}x_{n-3} \vee \bar{x}_{n-2}\bar{x}_{n-3}\cdots\bar{x}_{t+2}x_{t+1} \vee \bar{x}_{n-2}\bar{x}_{n-3}\cdots\bar{x}_{t+2}\bar{x}_{t+1}x_t x_{t-1} \vee\cdots\vee \bar{x}_{n-2}\cdots\bar{x}_1 x_0) = x_{n-1} \vee \bar{x}_{n-1}[x_{n-2} \vee x_{n-3} \vee \cdots \vee x_{t+1} \vee x_t(x_{t-1} \vee \cdots \vee x_0)]$,
$LE(n : 2^n-2) = \bar{x}_{n-1} \vee x_{n-1}[\bar{x}_{n-2} \vee \bar{x}_{n-3} \vee \cdots \vee \bar{x}_{t+1} \vee \bar{x}_t \vee x_t(\bar{x}_{t-1} \vee \cdots \vee \bar{x}_0)]$,
$RA(n : 1+2^t, 2^n-2) = \bar{x}_{n-1}[x_{n-2} \vee x_{n-3} \vee \cdots \vee x_{t+1} \vee x_t(x_{t-1} \vee \cdots \vee x_0)] \vee x_{n-1}[\bar{x}_{n-2} \vee \bar{x}_{n-3} \vee \cdots \vee \bar{x}_{t+1} \vee \bar{x}_t \vee x_t(\bar{x}_{t-1} \vee \cdots \vee \bar{x}_0)]$.
Consider the expression that involves variables from x_{n-1} to x_{t+1}:
$\bar{x}_{n-1}[x_{n-2} \vee x_{n-3} \vee \cdots \vee x_{t+1}] \vee x_{n-1}[\bar{x}_{n-2} \vee \bar{x}_{n-3} \vee \cdots \vee \bar{x}_{t+1}]$.
By Lemma 3.3, this expression can be simplified to one with fewer products. Next, consider the expression that involves the variable x_t:
$\bar{x}_{n-1}[x_{t-1} \vee x_{t-2} \vee \cdots \vee x_0] \vee x_{n-1}[\bar{x}_{t-1} \vee \bar{x}_{t-2} \vee \cdots \vee \bar{x}_0]$.
Again, by Lemma 3.3 this expression can be simplified to one with fewer products. Thus, in this case the function requires at most $2n-4$ products. In a similar way, we can show that the second case requires at most $2n-4$ products. Hence, we have the lemma. (Q.E.D.)

Up to here, we have shown that any range function can be represented by $2(n-2)$ products, when $n \geq 5$. From here, we are going to show that there exist range functions that require $2(n-2)$ products, when $n \geq 5$. To show the lower bound, we use an idea of independent sets of minterms [9].

Definition 3.3 *A set of minterms MI for f is* **independent** *if no implicant of f contains a pair of minterms in MI.*

Lemma 3.5 *Let MI be an independent set of minterms for f. Then, any SOP for f requires at least $|MI|$ products.*

Lemma 3.6 $\tau(RA(n : 2^{n-3}+1, 7\cdot 2^{n-3}-2)) = 2(n-2)$.

(Proof) $GE(n : 2^{n-3}+1) = x_{n-1} \vee \bar{x}_{n-1}x_{n-2} \vee \bar{x}_{n-1}\bar{x}_{n-2}x_{n-3}(x_{n-4} \vee x_{n-5} \vee \cdots \vee x_1 \vee x_0)$.
$LE(n : 7 \cdot 2^{n-3} - 2) = \bar{x}_{n-1} \vee x_{n-1}\bar{x}_{n-2} \vee x_{n-1}x_{n-2}\bar{x}_{n-3}(\bar{x}_{n-4} \vee \bar{x}_{n-5} \vee \cdots \vee \bar{x}_1 \vee \bar{x}_0)$.
$RA(n : 2^{n-3}+1, 7 \cdot 2^{n-3} - 2) = \bar{x}_{n-1}x_{n-2} \vee \bar{x}_{n-1}\bar{x}_{n-2}x_{n-3}(x_{n-4} \vee x_{n-5} \vee \cdots \vee x_1 \vee x_0) \vee x_{n-1}\bar{x}_{n-2} \vee x_{n-1}x_{n-2}\bar{x}_{n-3}(\bar{x}_{n-4} \vee \bar{x}_{n-5} \vee \cdots \vee \bar{x}_1 \vee \bar{x}_0)$
$= \bar{x}_{n-1}x_{n-2} \vee \bar{x}_{n-1}x_{n-3}(x_{n-4} \vee x_{n-5} \vee \cdots \vee x_1 \vee x_0) \vee x_{n-1}\bar{x}_{n-2} \vee x_{n-1}\bar{x}_{n-3}(\bar{x}_{n-4} \vee \bar{x}_{n-5} \vee \cdots \vee \bar{x}_1 \vee \bar{x}_0)$.
In the SOP that is obtained from the last expression contains $2(n-2)$ products. So, the function can be represented by $2(n-2)$ products.
Next, we will show that any SOP for the function requires at least $2(n-2)$ products. We will show this by showing the set of $2(n-1)$ independent minterms. Let MI be the set of minterms whose indices are $2^{n-3}+2^i$ and $7\cdot 2^{n-3}-1-2^i$, where $i \neq n-3$, and $i = 0, \ldots, n-2$. They corresponds to minterms of the functions, but no implicant of the function contains a pair of minterms in MI. (Q.E.D.)

Example 3.2 *Consider the case of $n = 6$. For $RA(n : 2^{n-3}+1, 7 \cdot 2^{n-3} - 2) = RA(6 : 9, 54)$, the vectors corresponding to the independent set are:*

$$\vec{a}_1 = (0,0,1,0,0,1),\ \vec{a}_2 = (0,0,1,0,1,0),$$
$$\vec{a}_3 = (0,0,1,1,0,0),\ \vec{a}_4 = (0,1,1,0,0,0),$$
$$\vec{a}_5 = (1,0,0,1,1,1),\ \vec{a}_6 = (1,1,0,0,1,1),$$
$$\vec{a}_7 = (1,1,0,1,0,1),\ \vec{a}_8 = (1,1,0,1,1,0).$$

Note that the integers represented by eight vectors are all in the range. Consider the pair $\vec{a}_1$ and $\vec{a}_2$. The products that contains the pair also contains the vector $(0,0,1,0,0,0)$. Note that this does not correspond to the implicant of the function, since the vector denotes the integer 8, which is out of range. Consider the pair $\vec{a}_1$ and $\vec{a}_5$. The products that contains the pair also contains the vector $(0,0,0,0,0,0)$. Again, this does not correspond to the implicant of the function, since the vector denotes the integer 0, which is out of range. Consider the pair $\vec{a}_7$ and $\vec{a}_8$. The products that contains the pair also contains the vector $(1,1,0,1,1,1)$. Again, this does not correspond to the implicant of the function, since the vector denotes the integer 55, which is out of range. (End of Example)

Example 3.3 *Consider the case where $n = 6$, $A = 2^{n-2} + 1 = 17$, and $B = 3 \cdot 2^{n-2} - 2 = 46$. In this case,*
$(a_5, a_4, a_3, a_2, a_1, a_0) = (0,1,0,0,0,1)$, and
$(b_5, b_4, b_3, b_2, b_1, b_0) = (1,0,1,1,1,0)$.
$GE(6 : 17) = x_5 \vee \bar{x}_5 x_4(x_3 \vee x_2 \vee x_1 \vee x_0)$.
$LE(6 : 46) = \bar{x}_5 \vee x_5\bar{x}_4(\bar{x}_3 \vee \bar{x}_2 \vee \bar{x}_1 \vee \bar{x}_0)$.
$RA(6 : 17, 46) = GE(6 : 17) \cdot LE(6 : 46)$
$= \bar{x}_5 x_4(x_3 \vee x_2 \vee x_1 \vee x_0) \vee x_5\bar{x}_4(\bar{x}_3 \vee \bar{x}_2 \vee \bar{x}_1 \vee \bar{x}_0)$.
Note that the SOP obtained from the above expression requires 8 products, and it is the minimum SOP. Thus, we can verify that $\tau(RA(6 : 17, 46)) = 2(n-2) = 8$. (End of Example)

Lemma 3.7 $\tau(RA(n : 2^{n-2}+1, 3\cdot 2^{n-2}-2)) = 2(n-2)$.

(Proof) $GE(n : 2^{n-2}+1) = x_{n-1} \vee \bar{x}_{n-1}x_{n-2}(x_{n-3} \vee x_{n-4} \vee \cdots \vee x_1 \vee x_0)$.
$LE(n : 3 \cdot 2^{n-2} - 2) = \bar{x}_{n-1} \vee x_{n-1}\bar{x}_{n-2}(\bar{x}_{n-3} \vee \bar{x}_{n-4} \vee \cdots \vee \bar{x}_1 \vee \bar{x}_0)$.
$RA(n : 2^{n-2}+1, 3\cdot 2^{n-2}-2) = \bar{x}_{n-1}x_{n-2}(x_{n-3} \vee x_{n-4} \vee \cdots \vee x_1 \vee x_0) \vee x_{n-1}\bar{x}_{n-2}(\bar{x}_{n-3} \vee \bar{x}_{n-4} \vee \cdots \vee \bar{x}_1 \vee \bar{x}_0)$.
In the SOP that is obtained from the last expression contains $2(n-2)$ products. So, the function can be represented by

$2(n-2)$ products.
Next, we will show that any SOP for the function requires at least $2(n-2)$ products. We will show this by showing the set of $2(n-1)$ independent minterms. Let MI be the set of minterms whose indices are $2^{n-2}+2^i$ and $3 \cdot 2^{n-2}-1-2^i$, where $i \neq n-2$, and $i = 0, \ldots, n-2$. They corresponds to minterms of the functions, but no implicant of the function contains a pair of minterms in MI. (Q.E.D.)

Example 3.4 *Consider the case of* $n = 6$. *For* $RA(n : 2^{n-2}+1, 3 \cdot 2^{n-2}-2) = RA(6 : 17, 46)$, *the vectors corresponding to the independent set are:*

$$\vec{b}_1 = (0,1,0,0,0,1),\ \vec{b}_2 = (0,1,0,0,1,0),$$
$$\vec{b}_3 = (0,1,0,1,0,0),\ \vec{b}_4 = (0,1,1,0,0,0),$$
$$\vec{b}_5 = (1,0,0,1,1,1),\ \vec{b}_6 = (1,0,1,0,1,1),$$
$$\vec{b}_7 = (1,0,1,1,0,1),\ \vec{b}_8 = (1,0,1,1,1,0).$$

(End of Example)

From Lemmas 3.4, 3.6 and 3.7, we have:

Theorem 3.2 $\mu(n) = 2(n-2)$, *where* $n \geq 5$.

Furthermore, we have the following:

Conjecture 3.1 *For* $n \geq 5$, *among the range functions, only*
$RA(n : 2^{n-3}+1, 7 \cdot 2^{n-3}-2)$ *and*
$RA(n : 2^{n-2}+1, 3 \cdot 2^{n-2}-2)$
require $2(n-2)$ *products, and other range functions require fewer products.*

Many people [12, 5, 4, 2, 13] in the network research believed that $\mu(n) = 2(n-1)$, however, it is not true. Some people [6] believed that $\mu(n) = 2n$, this also not true. **They considered the case where only the prefixes [1] are used to represent the function. However, by considering non-prefix-type products as well, we can improve the upper bound.** An important contribution of this paper is an improvement of an upper the bound on the size of SOPs for range functions.

4. Classification Functions

Definition 4.1 *A classification function with* n **fields** *is a mapping* $f : P_1 \times P_2 \times \cdots \times P_n \to \{0,1,2,\ldots,k\}$, *where* $P_i = \{0,1,\ldots,2^{t_i}-1\}$, *and each field is represented by* t_i *bits* $(i = 1,2,\ldots,n)$. f *is specified by a set of* k **rules**. *A rule consists of* n *fields, and is associated with an* **ID**.

Table 4.1. Range Match Table.

Rule #	X_1	X_2	ID
1	[0, 5]	[1,1]	1
2	[3,7]	[4,7]	2
3	[0,7]	[3,5]	3
4	[0,1]	[2,2]	4
5	[3,4]	[3,6]	5

Each field of a rule R has a **range** *specification. An input* $\vec{x}$ *matches a rule R, if the value of each field of* $\vec{x}$ *is in the range. Two distinct rules are either overlapping or non-overlapping, or that one is a subset of the other, with corresponding set-related definitions. When two rules have a common element, the order in which they appear in the classifier will determine their relative priority.*

Example 4.1 *Consider the* **range match table** *shown in Table 4.1. This table has two input fields:* X_1, *and* X_2. *Each filed has an interval* $[A, B]$, *where* A *is a lower bound, and* B *is an upper bound. Assume that the input pattern has a form* (a_1, a_2), *where* $a_i \in \{0,1,\ldots,7\}$. *When the input pattern is* $(0,1)$, *only the first rule matches. When the input pattern is* $(3,3)$, *the second, the third, and the fifth rules match. Since the second rule has the highest priority, the output is 2. In this example, Rule 4 and Rule 5 are disjoint, but Rule 2 and Rule 3 have common elements* [2]. *This is a* **range matching** *with* $n = 2$, $t_i = 3$ *and* $k = 5$. *(End of Example)*

Example 4.2 *Consider the Longest Prefix Match table (***LPM table***) shown in Table 4.2. For all the rules in the table, the asterisks (*) appear as the postfix only, if any. Also, the rules are sorted in the increasing order of asterisks. Note that an asterisk matches both 0 and 1, i.e., is a don't care. When the input pattern is* $(0,1,0,1)$, *the second, the third, and the fifth rules match. Since the second rule matches in the longest prefix, the output is 2. When the input pattern is* $(0,1,1,1)$, *the third and the fifth rules match. Since the third rule matches in the longest prefix, the output is 3. This is a* **longest prefix matching (LPM)**, *where* $n = 1$, $t_i = 4$, *and* $k = 5$. *(End of Example).*

An LPM is a special case of a range matching. For example, 01** is also denoted by the interval [4,7]. In an IP address look up, a *rule* corresponds to a *routing table entry*, a *range* corresponds to a *prefix*, an *action* corresponds to the *next-hop address*, and a *priority* corresponds to a *prefix-length* [3].

[1] In the network terminology, *prefix* corresponds to a logical product having a form $x_{n-1}^* x_{n-2}^* \cdots x_k^*$, where x_i^* denotes x_i or $\bar{x}_i$. For example, when $n = 4$, $x_3 x_2 \bar{x}_1$ is a prefix, while $x_3 \bar{x}_1$ is not.

[2] In Tables 4.1, 4.2, and, 4.3, the rule numbers and the IDs are the same. However, in the practical applications, different rules may have the same ID. In fact, in Table 6.1, Rules 6 and 8 share the same action

Table 4.2. LPM Table.

Rule #	Pattern	ID
1	1000	1
2	010*	2
3	01**	3
4	1***	4
5	0***	5

Table 4.3. Exact Match Table.

Rule #	Pattern	ID
1	0010	1
2	0111	2
3	1101	3
4	0101	4
5	0011	5
6	1011	6
7	0001	7

Example 4.3 *Consider the* **exact match table** *shown in Table 4.3. The output of the function is an ID if there is an exact match. When the input pattern is* $(1, 1, 0, 1)$*, since the third rule matches, the output is* 3*. When the input pattern is not stored in the table, the output is 0. This is an example of an* **exact matching** *[11, 10], where* $n = 4$ *and k=7.* *(End of Example)*

As shown in Examples 4.2 and 4.3, an exact matching is a special case of an LPM, where the range has the form $[A, A]$.

5. Complexity of Classification Functions

In this section, we consider the number of products to represent a classification function by using an SOP of binary variables. Let k be the number of rules. For exact matching and LPM, k products are sufficient to implement the circuit. However, for range matching, the necessary number of products can be much larger than k.

Theorem 5.1 *Consider the classification function* $f : P_1 \times P_2 \times \cdots \times P_n \rightarrow \{0, 1, \ldots, k\}$*, where* f *is specified by a set of* k *rules. Then* f *can be represented with at most* $k \cdot \prod_{i=1}^{n} \mu(t_i)$ *products, where* $P_i = \{0, 1, \ldots, 2_i^t - 1\}$*, and* t_i *is the number of bits to represent the i-th field* $(i = 1, 2, \ldots, n)$*.*

(Proof) By Theorems 3.1 and 3.2, each field requires at most $\mu(t_i)$ products. Thus, each rule can be represented with at most $\prod_{i=1}^{n} \mu(t_i)$ products by an SOP. Since, there are k rules, the function can be represented with most $k \cdot \prod_{i=1}^{n} \mu(t_i)$ products. (Q.E.D.)

Example 5.1 *Consider the classification function defined by Table 4.1. Note that* $n = 2$*,* $t_1 = t_2 = 3$*,* $k = 5$*, and* $\mu(3) = 3$*. The function requires at most* $5 \times 3 \times 3 = 45$ *products.* *(End of Example)*

Example 5.2 *Consider the classification function defined in the introduction. In this case,* $n = 2$*,* $t_1 = t_2 = 2$*,* $k = 2$*,*

Table 5.1. TCAM Pattern.

x_1	x_2	x_3	x_4	Index
0	$*$	1	$*$	1
0	$*$	$*$	1	2
$*$	0	1	$*$	3
$*$	0	$*$	1	4
1	$*$	0	$*$	5
1	$*$	$*$	0	6
$*$	1	0	$*$	7
$*$	1	$*$	0	8

Table 5.2. RAM Pattern.

y_1	y_2	y_3	y_4	Index
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	2
0	1	1	0	2
0	1	1	1	2
1	0	0	0	2

and $\mu(2) = 2$*. Theorem 5.1 shows that this function can be represened with at most* $k\mu(t_1)\mu(t_2) = 2 \cdot 2 \cdot 2 = 8$ *products. In fact, this function requires 8 products as follows:* $R_1 = (\bar{x}_1 \vee \bar{x}_2)(x_3 \vee x_4) = \bar{x}_1 x_3 \vee \bar{x}_1 x_4 \vee \bar{x}_2 x_3 \vee \bar{x}_2 x_4$*.* $R_2 = (x_1 \vee x_2)(\bar{x}_3 \vee \bar{x}_4) = x_1 \bar{x}_3 \vee x_1 \bar{x}_4 \vee x_2 \bar{x}_3 \vee x_2 \bar{x}_4$*. To implement this function, we use the TCAM shown in Table 5.1, and the RAM shown in Table 5.2. In the TCAM, the upper four words detect Rule 1, while the lower four words detect Rule 2. The RAM converts the index of the TCAM to the index of the rules.* *(End of Example)*

Note that the cost of TCAM bit is much higher than that of RAM [13], so we can virtually ignore the cost of the RAM.

6. Packet Classification Functions

In the packet classification, each rule specifies possible values of the source, destination addresses of the IP header, the protocol field, and the source and destination port numbers (for TCP and UDP). The address fields are often specified by prefixes of IP addresses. The port fields are specified by ranges of port numbers. Protocols are either specified exactly or as a wildcard [12].

Example 6.1 *Table 6.1 shows an example of a rule set for a packet classification. It has five input fields: Source Address, Destination Address, Protocol, Source Port, and Destination Port, denoted by* X_1*,*X_2*,*X_3*,*X_4*, and* X_5*, respectively. Note that address fields have only four bits, rather than 32 bits to simplify the example. Also, port fields have only four bits rather than 16 bits. The protocol field has three values: TCP, UDP, and ICMP. An asterisk in an address field indicates a bit position that can be either 0 or 1. A dash for an entry denotes a wildcard that matches any pattern. When a packet is received, the classifier finds the first matching rule in the set. The packet is then processed according to the specified action. For example, a packet with the source address 0101, the destination address 0011, the protocol field specifying TCP, the source*

Table 6.1. Packet Classification Table.

Rule No.	Sour. Addr.	Dest. Addr.	Prot.	Sour. Port	Dest. Port	Act.
	X_1	X_2	X_3	X_4	X_5	f
1	11**	01**	TCP	[2,4]	[7,7]	fwd 3
2	01**	00**	TCP	[3,9]	[2,6]	fwd 5
3	110*	101*	UDP	[1,7]	[4,6]	fwd 2
4	0101	11**	ICMP	[0,15]	[0,15]	fwd 7
5	001*	011*	UDP	[4,4]	[5,5]	fwd 0
6	111*	10**	-	[0,15]	[0,15]	drop
7	0101	****	-	[0,15]	[0,15]	fwd 4
8	1***	0***	-	[0,15]	[0,15]	drop

port number 4 and the destination port number 6 would be forwarded to output port 5, since it matches the second rule in the set, but not the first. On the other hand, a packet with the source address 1111 and the destination address 1000 would be dropped regardless of other fields, since the first matching rule is number 6. In this example, X_1 and X_2 take 16 values, X_3 takes 3 values, X_4 and X_5 take 16 values. *(End of Example)*

A practical packet header has 8 fields, uses 208 bits in total.

7. Conclusion and Comments

In this paper, we first defined range functions as a generalization of comparator functions. Second, we derived an upper bound on the number of products in the SOP to represent a range function. Third, we defined classification functions, which is a generalization of packet classification functions. These bounds are useful to estimate the amount of hardware to implement classification functions. Especially, we have improved an upper bound on the number of products to represent range functions.

Acknowledgments

The author appreciates the reviewers who patiently read the paper, and pointed out flaws of the original manuscript. This research is supported in part by the Grants in Aid for Scientific Research of JSPS, and the grant of the MEXT Knowledge Cluster Project.

References

[1] F. Baboescu and G. Varghese, “Scalable packet classification,” *IEEE/ACM Transactions on Networking*(TON), Vol.13, No.1, pp.2-14, Feb. 2005.

[2] Q. Dong, S. Banerjee, J. Wang, D. Agrawal, and A. Shukla, “Packet classifiers in ternary CAMs can be smaller,”*ACM SIGMETRICS 2006*, pp.311-322.

[3] P. Gupta and N. McKeown, “Packet classification on multiple fields,” *Proc. of ACM SIGCOMM*, pp. 147-160, 1999.

[4] P. Gupta and N. McKeown, “Algorithms for packet classification,” *IEEE Network*, Special Issue, March/April 2001, Vol. 15, No. 2, pp 24-32.

[5] K. Lakshminarayanan, A. Rangarajan, and S. Venkatachary, “Algorithms for advanced packet classification with ternary CAMs,” *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, August 22-26, 2005, Philadelphia, Pennsylvania, USA.

[6] H. Liu,“Efficient mapping of range classifier into ternary-CAM,” *10th Symposium on High Performance Interconnects HOT Interconnects* (HotI’02), pp. 95-100, 2002.

[7] K. Pagiamtzis and A. Sheikholeslami, “Content-addressable memory (CAM) circuits and architectures: A tutorial and survey,” *IEEE Journal of Solid-State Circuits*, vol. 41, No. 3, pp. 712-727, March 2006.

[8] T. Sasao, *Switching Theory for Logic Synthesis*, Kluwer Academic Publishers, 1999.

[9] T. Sasao and J. T. Butler,“Worst and best irredundant sum-of-products expressions,”*IEEE Transactions on Computers*, Vol. 50, No. 9,Sept. 2001, pp.935-948.

[10] T. Sasao and J. T. Butler,“Implementation of multiple-valued CAM functions by LUT cascades,” *ISMVL-2006*, Singapore, May 17-20, 2006.

[11] T. Sasao, “Design methods for multiple-valued input address generators,”(invited paper) *International Symposium on Multiple-Valued Logic* (ISMVL-2006), Singapore, May 2006.

[12] E. Spitznagel, D. Taylor, and J. Turner, “Packet classification using extended TCAMs,” *Proc. 11th IEEE International Conference on Network Protocols*, (ICNP 03), Nov.4-7, 2003, pp. 120-131.

[13] D. E. Taylor,“Survey and taxonomy of packet classification techniques,” *ACM Computing Surveys*, Vol.37, Issue 3, Sept. 2005,pp. 238 - 275.

[14] F. Yu, R.H. Katz, and T.V. Lakshman, “Efficient multimatch packet classification and lookup with TCAM,” *IEEE Micro*, Vol.25, No.1, pp.50-59, Jan.- Feb. 2005.