TCAM representations of intervals of integers encoded by binary trees*

Wojciech Fraczak[1], Wojciech Rytter [2,3]
Mohammadreza Yazdani[4]
[1]Dépt d'informatique, Université du Québec en Outaouais
Gatineau PQ, Canada
[2]Inst. of Informatics, Warsaw University
Warsaw, Poland
[3]Department of Mathematics and Informatics
Copernicus University, Torun, Poland
[4]Systems and Computer Engineering, Carleton University
Ottawa ON, Canada

Abstract

We investigate the problem of efficient representations of intervals of positive integers in TCAM (Ternary Content Addressable Memory). The integers are encoded by binary strings of the same length n and a TCAM of width n is a string-oriented representation of arbitrary sets of n-bit strings in terms of a collection of simple sets, called rules. Each rule is a concatenation (of length n) of singleton sets (i.e., single digits 0 and 1) or the set $\{0,1\}$ denoted by $*$. We consider a family of n-bit encodings for integers, called dense-tree encodings, which includes the lexicographic encoding (i.e., standard unsigned binary encoding) and the binary reflected Gray encoding. We provide exact bounds (with respect to n) on the minimal sizes of TCAMs representing a subset of n-bit strings corresponding to an interval. Some other issues related to the minimal sizes and number of *essential* rules of TCAMs are also investigated.

*The research of the first, second, and the third author were supported by grants of NSERC, Polish Ministery of Science and Higher Education N 206 004 32/0806, and Ontario Graduate Scholarship, respectively.

1 Introduction

The *Ternary Content Addressable Memory* (TCAM), [7, 8], is a type of associative memory with a highly parallel architecture which is used for performing very fast (constant time) table look up operations. The problem of interval representation by TCAM appears in network processing engines where the header fields of each IP packet (e.g., source address, destination address, port number, etc.) should be matched under strict time constraints against the entries of an Access Control List (ACL) [2, 6, 11]. Often, an ACL is implemented in TCAM which allows for matching a given IP header against all ACL entries in parallel. Each entry of the ACL defines either a single value or an interval of values for the fields of the packet header. If an ACL entry defines only single values for all header fields, then it can be directly and very efficiently represented using a single TCAM rule. However, if the ACL entry defines some non-trivial intervals for some header fields of the packet, then it may need more than one TCAM rule [9, 10].

The problem of an efficient representation of intervals in TCAM is sometimes referred to as the problem of "range representation" in TCAM, [10, 9, 2]. In this paper we present a systematic approach for tackling this problem by focusing on efficient representation of a single interval in TCAM. For a family of encoding schemes, which includes the lexicographic encoding (i.e., standard unsigned binary encoding) and the binary reflected Gray encoding, we provide exact bounds on the minimal sizes of TCAMs for interval representation. We also provide bounds on the number of *essential* TCAM rules (prime implicants) for intervals in the cases of the lexicographic encoding and the binary reflected Gray encoding.

2 Definition of the problem

A TCAM can be defined as a two dimensional array of cells, where each cell carries one of the three values 0, 1, or $*$. Each row of this array is called a TCAM rule. Examples of TCAMs are shown in Figure 1. A TCAM rule is called a prefix rule if all non-star symbols occur as a prefix of it, e.g., 0101**** is a prefix rule. The TCAM in Figure 1-(a) uses exclusively prefix rules.

A rule of a TCAM of width n is a sequence $r = e_1 e_2 \dots e_n$, where $e_i \in \{0, 1, *\}$ for $i \in \{1, \dots, n\}$. It defines the following non-empty language $L(r)$:

$$L(r) \stackrel{\text{def}}{=} L(e_1)L(e_2)\dots L(e_n),$$

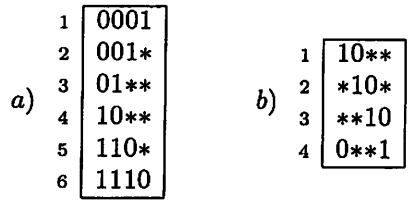

Figure 1: TCAM representations for the interval $R = [1, 14]$, where the integers are represented in the standard binary encoding over 4 bits.

where $L(0) \stackrel{\text{def}}{=} \{0\}$, $L(1) \stackrel{\text{def}}{=} \{1\}$, and $L(*) \stackrel{\text{def}}{=} \{0, 1\}$. For example, $L(0*1) = \{0\} \cdot \{0, 1\} \cdot \{1\} = \{001, 011\}$. We say that r *covers* a set of strings S if $S \subseteq L(r)$.

TCAM $\mathcal{R}$ consisting of k rules $r_1, \ldots, r_k$ will be written as $\mathcal{R} = (r_1, \ldots, r_k)$. The language of $\mathcal{R} = (r_1, \ldots, r_k)$ is the union of the languages defined by its rules, i.e., $L(\mathcal{R}) \stackrel{\text{def}}{=} L(r_1) \cup \ldots \cup L(r_k)$.

Let $\mathcal{R} = (r_1, r_2, \ldots, r_k)$ and $\mathcal{R}' = (p_1, p_2, \ldots, p_m)$ be two TCAMs of the same width d, i.e., $r_i, p_j \in \{0, 1, *\}^d$ for $i \in \{1, 2, \ldots, k\}$ and $j \in \{1, 2, \ldots, m\}$, and $w \in \{0, 1, *\}^n$ be a rule of length n. We define:

$$\begin{aligned} \mathcal{R} + \mathcal{R}' &\stackrel{\text{def}}{=} (r_1, r_2, \ldots, r_k, p_1, p_2, \ldots, p_m) \\ w \cdot \mathcal{R} &\stackrel{\text{def}}{=} (wr_1, wr_2, \ldots, wr_k) \ . \end{aligned}$$

Intuitively, $\mathcal{R} + \mathcal{R}'$ is the union of $\mathcal{R}$ and $\mathcal{R}'$, thus its width remains d and the number of its rules is $k + m$. TCAM $w \cdot \mathcal{R}$ is of width $d + n$ and each of its rules is the concatenation of w with a rule of $\mathcal{R}$.

Let $E : \{0, 1\}^n \hookrightarrow \{0, 1, \ldots, 2^n - 1\}$ be an encoding of integer values by n-bit strings. Any subset $X \subseteq \{0, 1, \ldots, 2^n - 1\}$ can be represented by a TCAM $\mathcal{R}$ of width n, i.e., $X = E(L(\mathcal{R}))$.

The problem of finding a minimum size TCAM $\mathcal{R}$ (i.e., a TCAM with the minimum number of rules) for a given set X is known to be NP-hard (as it corresponds to the problem of finding a minimal disjunctive normal form for a Boolean expression). However, in this paper we are interested in very special sets of binary strings which correspond to the binary encodings of the numbers in a given *interval*. An interval $[x, y]$ is the set of all integer between the two end points x and y, i.e., $[x, y] = \{x, x + 1, \ldots, y\}$. An interval can be represented by several different TCAMs. For example, if we use the lexicographic encoding to represent the integers with 4 bits, then Figure 1 shows two different TCAM representations for the interval

[1, 14]. In the case of the lexicographic encoding, the most widely-used way of representing intervals in TCAM is called "prefix rule TCAM" and consists in dividing a given interval into a number of non-overlapping sub-intervals such that each sub-interval can be represented by a single prefix rule. The TCAM representation is then generated as the union of all the prefix rules corresponding to these sub-intervals. While this algorithm is simple, fast, and generates the unique (up to a permutation of the rules) result, it compromises on the number of the rules in the generated solution. For example, for the interval [1, 14] the prefix rule TCAM, as shown in Figure 1-(a), has two rules more than the TCAM representation of Figure 1-(b).

3 Dense-tree encodings of integers and interval sets

We define a *full* tree as a perfect binary tree such that each pair of sibling edges are labeled 0 and 1. The assignment of labels to the edges (two alternatives per each internal node) can be chosen arbitrarily.

Let T_n be a full tree of height n. The label $w \in \{0,1\}^n$ of the path from the root to i-th leaf defines the n-bit encoding of number i, with 0 corresponding to the furthest left leaf of the tree. In that way, T_n defines a bijection $T_n : \{0,1\}^n \hookrightarrow \{0, 1, \ldots, 2^n - 1\}$, called an n-bit *dense-tree encoding*. The lexicographic encoding and the binary reflected Gray encoding (see [3], [5]) are two important examples of dense-tree encodings. They are presented in Figure 2 in forms of full trees for 4-bit encodings.

In the context of a dense-tree encoding T_n, a set $X \subseteq \{0,1\}^n$ defines both the set $T_n(X)$ of integers and a subset of leaves of T_n. X can be represented by a *skeleton tree* (see Figure 3). The skeleton tree of X is obtained from T_n by:

(1) removing all edges which are not leading to the leaves of X;

(2) turning into leaves all full sub-trees.

We say that a skeleton tree S is a *chain*, if every vertex of S has at most one non-leaf child. A *double-chain* is a skeleton tree with at most one vertex v having two non-leaf children and such that all ancestors of v have only one child. Examples of a chain and a double-chain are illustrated in Figure 4.

A set of binary strings X of length n is an *interval-set* of a dense-tree encoding T_n if $T_n(X)$ is an interval.

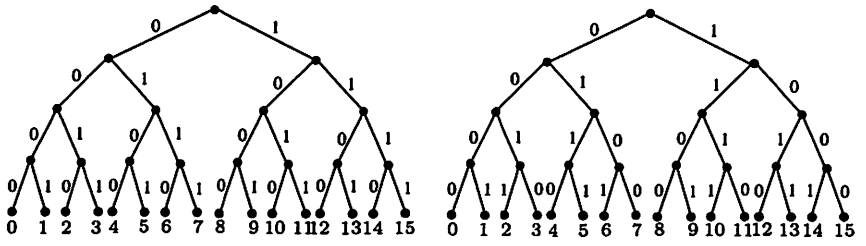

Figure 2: Two sample 4-bit dense-tree encodings: Lexicographic encoding (Lex) and binary reflected Gray encoding (Gray). In the case of the binary reflected Gray encoding every pair of sibling sub-trees are labeled symmetrically, thus the full labeling of the tree is determined by the label of any of its branches.

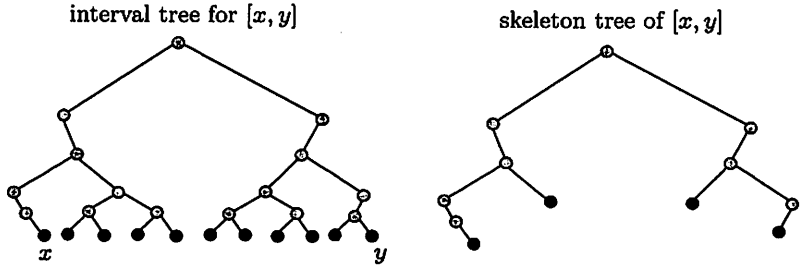

Figure 3: The interval tree and skeleton tree of an interval $[x, y]$.

Lemma 1 *Let $X \subseteq \{0,1\}^n$. The following statements are equivalent:*

1. *There exists a dense-tree encoding T_n for which $T_n(X) = [0, |X| - 1]$;*
2. *There exists a dense-tree encoding T_n for which $T_n(X) = [2^n - |X|, 2^n - 1]$;*
3. *For every dense-tree encoding, the skeleton tree of X is a chain.*

Proof. Equivalence $1 \Leftrightarrow 2$ and implication $3 \Rightarrow 1$ are obvious. We present a proof based on induction on n for $1 \Rightarrow 3$. For $n = 1$ any subset of $\{0,1\}$ is an interval, and any skeleton tree of height not bigger than 1 is a chain.

Suppose that the induction hypothesis holds for all $k \leq n$. For $X \subseteq \{0,1\}^{n+1}$, we can write $X = X_0 \cup X_1$, where X_0 and X_1 are the sets of all those strings of X which start by 0 and 1, respectively. There are two cases:

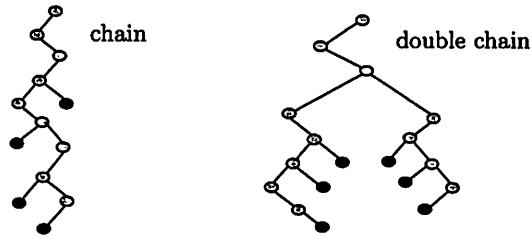

Figure 4: A chain and a double-chain.

1. If X_0 or X_1 is empty, then the initial symbol is the same for all elements of X and it can be omitted. Thus, the proof is complete as the induction hypothesis applies directly.

2. Suppose that none of X_0 or X_1 is empty and consider an arbitrary dense-tree encoding T_{n+1}. Since by 1) there exists a dense-tree encoding T'_{n+1} for which $T'_{n+1}(X) = [0, |X| - 1]$, we must have $|X| > 2^{n-1}$ and also either X_0 or X_1 must include 2^{n-1} strings. We assume without loss of generality that $|X_0| = 2^{n-1}$. The skeleton tree of X_0 in T_{n+1} is just an edge labelled by 0, and, by the induction hypothesis, the skeleton tree of X_1 is a single edge connected to a chain. Thus, the root of the skeleton tree of X in T_{n+1} has a leaf as one child and a chain connected to the other child, i.e., it is a chain. □

Lemma 2 *Let $X \subseteq \{0,1\}^n$. The following statements are equivalent:*

1. *There exists a dense-tree encoding T_n for which $T_n(X)$ is an interval;*

2. *For every dense-tree encoding the skeleton tree of X is a double-chain.*

Proof. The proof of $2 \Rightarrow 1$ is obvious. We prove the $1 \Rightarrow 2$ part by induction on n. For $n = 1$ see the proof of Lemma 1.

Suppose that the induction hypothesis holds for all $k \leq n$. For $X \subseteq \{0,1\}^{n+1}$, we can write $X = X_0 \cup X_1$, where X_0 and X_1 are the sets of all those strings of X which start by 0 and 1, respectively. There are two cases:

1. If X_0 or X_1 is empty, then the initial symbol is the same for all elements of X and it can be omitted. Thus, the proof is complete as the induction hypothesis applies directly.

2. Suppose that none of X_0 or X_1 is empty and consider an arbitrary dense-tree encoding T_{n+1}. Since there exists a dense-tree encoding

T'_{n+1} for which $T'_{n+1}(X)$ is an interval, by Lemma 1, the skeleton trees of X_0 and X_1 in T_{n+1} are chains. Thus, only the root of the skeleton tree of X in T_{n+1} has two non-leaf children which means it is a double chain. □

Given a skeleton tree of a set $X \in \{0,1\}^n$, one can check if the tree is a double-chain in time $O(n)$. Therefore, we have the following corollary from the result of Lemma 2:

Corollary 2.1 *Given a skeleton tree of a set $X \in \{0,1\}^n$, there is a linear time algorithm to test whether there exists a dense-tree encoding T_n for which $T_n(X)$ is an interval.*

In contrast the problem of determining whether a given TCAM defines an interval is intractable, as stated by the following fact.

Theorem 3 *The problem of testing for a given TCAM $\mathcal{R}$ if $L(\mathcal{R})$ is an interval set in a given dense-tree encoding is co-NP hard.*

Proof. Testing if a TCAM of width n corresponds to a full tree is co-NP hard, since determining whether a given Boolean formula in a disjunctive normal form is a tautology is known to be co-NP complete. For a given TCAM $\mathcal{R}$, we construct the TCAM $\mathcal{R}' = \{10\} \cdot \mathcal{R} + \{01^{n+1}\} + \{10^{n+1}\}$. TCAM $\mathcal{R}'$ defines an interval set in the lexicographic encoding if and only if $\mathcal{R}$ corresponds to a full tree. □

4 Prefix and essential TCAM rules

A TCAM rule $r \in \{0,1,*\}^n$ is called a prefix rule if all non-star symbols occur as a prefix of it, i.e., $r \in \{0,1\}^k *^{n-k}$, for some $k \in \{0,\ldots,n\}$.

Lemma 4 *Let $X \subseteq \{0,1\}^n$ be an interval-set of a dense-tree encoding T_n. The minimum number of prefix rules covering X equals the number of leaves in the skeleton tree of X in T_n.*

Proof. Each prefix rule corresponds to the path from the root to a leaf with all remaining symbols set to $*$. The rules are disjoint and any other prefix rule for X is strictly included in one of those prefix rules. □

Theorem 5 *Let the skeleton tree of $X \subseteq \{0,1\}^n$ in a dense-tree encoding T_n be a chain with k leaves. The minimal size of a TCAM for X is k.*

Proof. We first show that k rules are sufficient for representing X. Let k denote the number of leaves. Obviously k prefix rules are enough, each rule corresponding to a leaf.

Now we show that we need at least k rules (prefix or non-prefix). Assume, by contradiction, that we have k' rules $e_1, e_2, \ldots, e_{k'}$, with $k' < k$. Let $d = d[1]d[2]\ldots d[n]$ be the label of the path from the root to the sibling of the leftmost leaf of the interval on its full tree, which means that $d \notin X$. Each rule e_i has to have at least one position p_i such that $e_i[p_i]$ is $\overline{d[p_i]}$; we choose always the first such position. Since $k' < k$, there is a position r which corresponds to an incoming edge of a leaf of the chain and which was not selected by any of positions p_i. We change the symbol on this position of d to $\overline{d[r]}$. Then the resulting string belongs to the set X but it is not covered by any of the rules. □

Corollary 5.1 *In any n-bit dense-tree encoding, representing each of the intervals $[1, 2^n - 1]$ and $[0, 2^n - 2]$ needs exactly n TCAM rules.*

Consider a set $X \subseteq \{0,1\}^n$. A TCAM rule r is called an *X-limited* rule if it does not cover any string out of X, i.e., $L(r) \subseteq X$. An X-limited rule r is said to be *X-essential* if there is no other X-limited TCAM rule that covers $L(r)$. Any coverage of X by a TCAM of size k can be turned into a coverage by k X-essential TCAM rules. Therefore, one can consider only X-essential TCAM rules in the process of finding a minimal coverage of X.

In the context of two-level logic generation, an X-essential TCAM rule is called a "prime implicant" of X (see [1]). Prime implicants play an important role in the process of two-level logic generation. In general, the number of prime implicants for a given set may be exponential with respect to the number n of input bits [1]. In the following, we prove that in the case of the n-bit lexicographic encoding and n-bit reflected Gray encoding, an interval-set $X \subseteq \{0,1\}^n$ admits no more than $n(n-1)+1$ and $2n$ X-essential TCAM rules, respectively.

Lemma 6 *Let $X \subseteq \{0,1\}^n$ such that the skeleton tree of X in a dense-tree encoding T_n is a chain with k leaves. There are exactly k different X-essential TCAM rules.*

Proof. By induction on n. For $n = 1$ the lemma is obviously true.

Let C be a chain in T_{n+1} with k leaves.

If the root of C is a leaf than there is only one essential rule, $*^{n+1}$.

If the root of C has one child C_a (a chain in T_n) with its incoming edge labeled $a \in \{0,1\}$, then number of leaves of C and C_a is the same as well as

the number of the essential rules. The essential rules for C are the essential rules for C_a prefixed by a.

If the root of C has two children C_0 and C_1 (chains in T_n), then one of them is a leaf and the other has $k-1$ leaves. Thus, if C_a, for $a \in \{0,1\}$, is a leaf then $a*^n$ is an essential rule for C. Also, $*r$ is an essential rule for C if and only if r is an essential rule for $C_{\overline{a}}$. By the induction hypothesis, the number of the essential rules for $C_{\overline{a}}$ is $k-1$, therefore the number of the essential rules for C is $1+(k-1)=k$. □

Theorem 7 *Let $X \subseteq \{0,1\}^n$ be an interval-set in the n-bit lexicographic encoding $\mathtt{Lex}_n$. There is at most $n(n-1)+1$ different X-essential TCAM rules.*

Proof. Let $\mathtt{Lex}_n(X) = [x,y]$. The proof is by induction on n. Suppose that $\mathtt{Lex}_n(0u) = x$ and $\mathtt{Lex}_n(1v) = y$ for some $u,v \in \{0,1\}^{n-1}$. The easy case when x and y encodings are starting by the same digit is skipped.

The set $P([\mathtt{Lex}_n(0u), \mathtt{Lex}_n(1v)])$ of all X-essential TCAM rules can be split into three disjoint sets P_0, P_1, and P_*, where P_a, $a \in \{0,1,*\}$, denotes the set of all X-essential TCAM rules that start with a. By Lemma 6, we have: $|P_0| \leq n-1$ and $|P_1| \leq n-1$. Also $|P_*| = |P([\mathtt{Lex}_n(u), \mathtt{Lex}_n(v)])|$, where the interval $[u,v]$ is encoded by $n-1$ bits. Thus, $p(n) \leq 2(n-1)+p(n-1)$, with $p(1)=1$, where $p(i)$ denotes the maximum number of different X-essential TCAM rules for any interval I with $\mathtt{Lex}_i(X) = I$. □

Theorem 8 *Let $X \subseteq \{0,1\}^n$ be an interval-set in the reflected Gray encoding $\mathtt{Gray}_n$. There is no more than $2n$ different X-essential TCAM rules.*

Proof. Let $\mathtt{Gray}_n(X) = [x,y]$ be such that encodings of x and y differ in the first digit. The easy case when the encodings of x and y begin by the same digit is skipped. In reflected Gray encoding, the dense-tree is "reflective", i.e., the labels of the right-hand side and the left-hand side subtrees rooted at every node, are the mirror copies of each other. Suppose that $x < y$, x is encoded as av, y as $\overline{a}u$, $a \in \{0,1\}$, and $u,v \in \{0,1\}^{n-1}$. Then by reflective property, au encodes $2^n - 1 - y$.

In case when $x \leq 2^n - 1 - y$, there is no X-essential TCAM rule which starts by $\overline{a}$, and symmetrically, if $x \geq 2^n - y$ then there is no X-essential TCAM rule which starts by a. Without loss of generality we may assume that $x \leq 2^n - 1 - y$. Therefore, X-essential TCAM rules can be split into two sets P_a (i.e., the set of rules starting by a), and P_* (i.e., those rules that start by $*$). Every rule from P_a with initial a removed must be an essential TCAM rule for interval $[x, 2^{n-1}-1]$ on $n-1$ bits, and every rule

from P_* with initial $*$ removed must be an essential TCAM rule for interval $[2^n - 1 - y, 2^{n-1} - 1]$ on $n-1$ bits. Since the skeleton trees of both these intervals are chains (or empty), by Lemma 6 each of them has at most $n-1$ (or none if empty) different essential TCAM rules. □

5 Bounds on the size of a TCAM representing an interval

We first show that, for all n-bit dense-tree encoding schemes, any interval over $D = [0, 2^n - 1], n \geq 1$, can be represented by $\max(n, 2n-3)$ or less TCAM rules. We then show that in cases of the lexicographic and the binary reflected Gray encodings, the maximum number of rules is actually $2n-4$.

As it was proved in Lemma 4, the minimum number of prefix rules required for representing an interval in TCAM is equal to the number of leaves in its corresponding skeleton tree. For some intervals, this number can be significantly reduced by using non-prefix TCAM rules.

Theorem 9 *There is a dense-tree encoding T_n and an interval I such that the minimum number of prefix rules representing I is $2n-2$ and the minimum TCAM size is at most n.*

Proof. Consider the interval $I = [1, 2^n - 2]$ in the lexicographic encoding on n bits. The skeleton tree for $X \subset \{0,1\}^n$ such that $\mathtt{Lex}_n(X) = I$ has $2n-2$ leaves and thus by Lemma 4 it cannot be covered by less than $2n-2$ prefix rules. However X can be covered by the following n rules:

$$X = \{rot^k(01*\ldots*) \mid 0 \leq k < n\},$$

where rot denotes the (right-)rotation of a word, i.e.,

$$rot(wa) \stackrel{\text{def}}{=} aw, \text{ for } a \in \{0,1,*\} \text{ and } w \in \{0,1,*\}^{n-1},$$

(see Figure 1-(b) for an example when $n = 4$). □

Theorem 10
(a) *Let T_n be a lexicographic or reflected Gray encoding of length n. There is an interval $I = [x, y]$ which cannot be represented with less than* $\max(n, 2n-4)$ *TCAM rules.*
(b) *For each $n \geq 1$ there exists a dense-tree encoding T_n and an interval $I = [x, y]$ which needs* $\max(n, 2n-3)$ *TCAM rules.*

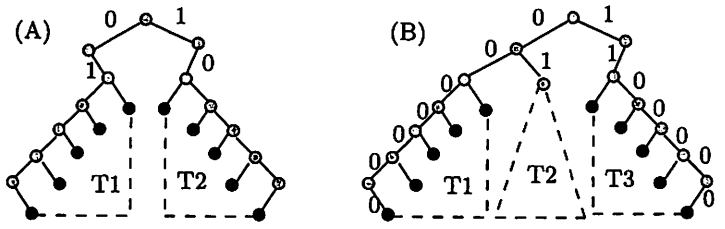

Figure 5: The lexicographic tree in (A) needs at least $2n-4$ rules for lexicographic coding, the (non-lexicographic) tree in (B) needs at least $2n-3$ TCAM rules.

Proof.
Point (a). For $1 \leq n \leq 3$, we have $\max(n, 2n-4) = n$ and the theorem's claim can be easily proved using Corollary 5.1.
For $n \geq 4$, we provide only a proof for the lexicographic encoding; the proof for the reflected Gray encoding is similar to this proof. Consider the interval

$$I = [r_l, r_h] = [\mathrm{Lex}_n(s_l), \mathrm{Lex}_n(s_h)] = [\mathrm{Lex}_n(0100\ldots001), \mathrm{Lex}_n(1011\ldots110)],$$

whose corresponding interval-set has a skeleton tree as illustrated in Fig. 5(A) for the lexicographic encoding. s_l starts with 01, followed by $n-3$ zeros and ends by 1. s_h starts with 10, followed by $n-3$ ones and ends by 0. We divide this interval to two sub-intervals

$$I_1 = [r_l, 2^{n-1}-1] = [\mathrm{Lex}_n(0100\ldots001), \mathrm{Lex}_n(0111\ldots11)],$$

$$I_2 = [2^{n-1}, r_h] = [\mathrm{Lex}_n(100\ldots00), \mathrm{Lex}_n(1011\ldots110)].$$

The encodings of all values in the interval I_1 start with 01 and thus they are within the first half of the domain $[0, 2^n-1]$. This means that all the leaves corresponding to the values in I_1 fall on the left side of the vertical line that passes through the root of the tree.

The encodings for the values of interval I_2, however, start with 10 and so their corresponding leaves fall on the right side of the vertical line. This means that any TCAM rule that represents a value from I_1 cannot represent any value from I_2 and vice verse. As such, the minimum number of TCAM rules required for representing I is equal to the addition of the minimum number of rules that are required to represent each of the intervals I_1 and I_2, separately.

Since the encodings of all the values from I_1 start with 10, we can represent this interval by prepending 10 to the $n-2$ rules that are required

to represent the interval $[1, 2^{n-2}-1] = [\mathtt{Lex}_n(00\ldots001), \mathtt{Lex}_n(11\ldots11)]$ over the domain $[0, 2^{n-2}-1]$.

Hence, using Corollary 5.1, we need $n-2$ TCAM rules of width $n-2$ to represent this interval. Similarly, we need $n-2$ TCAM rules for I_2. Thus, the interval I cannot be represented by less than $2n-4$ TCAM rules.

Point (b). An example of such a dense-tree encoding T_n and the interval $[1, 2^{n-1}+2^{n-2}-2]$ is shown in Figure 5(B).

No string starting by 10 is in the set, therefore only $*1$, $0*$, and 01 are possible prefixes for the essential rules of the interval. One can check using similar arguments as in Point (a), that we need exactly $n-2$ rules to cover $T1$, another $n-2$ rules to cover $T3$, and an additional rule to cover $T2$ since $010\ldots01$ is not covered by any rules intersecting with $T1$ or $T2$. This completes the proof. □

Theorem 11 *Let T_n be a dense-tree encoding of length n. Every interval $I = [x, y]$ can be represented by* $\max(n, 2n-3)$ *TCAM rules.*

Proof. If $1 \leq n \leq 3$, we have $\max(n, 2n-3) = n$ and it can be checked manually that every interval can be represented by n or less TCAM rules. For $n > 3$, if the skeleton tree of I has $2n-3$ or less leaves, then by Lemma 4, I can be represented by $2n-3$ or less prefix TCAM rules. The skeleton tree has maximal number of $2n-2$ leaves if and only if it has a shape similar to tree T of Figure 6. We decompose this tree into three sub-trees T_1, T_2, and T_3. The sub-trees T_1 and T_2 are chains with $n-3$ leaves each, and thus altogether they need $2n-6$ TCAM rules. It is enough to show that T_3 needs at most 3 TCAM rules. Let abc $(\overline{a}b'c')$ be the labels of the left (resp., right) branch of T_3, as shown in Figure 6. We consider all possible cases as follows. If $b'c' = \overline{b}\overline{c}$, then rules $\mathtt{a}*\mathtt{c}$, $*\overline{\mathtt{b}}\overline{\mathtt{c}}$, and $\overline{\mathtt{a}}\mathtt{b}*$ represent T_3; If $b'c' = \overline{b}c$, then rules $**\mathtt{c}$, $\mathtt{a}\overline{\mathtt{b}}\overline{\mathtt{c}}$, and $\overline{\mathtt{a}}\mathtt{b}\overline{\mathtt{c}}$ represent T_3; If $b'c' = bc$, then rules $*\mathtt{bc}$ and $*\overline{\mathtt{b}}*$ represent T_3; If $b'c' = b\overline{c}$, then rules $*\overline{\mathtt{b}}*$, $\mathtt{a}*\mathtt{c}$, and $\overline{\mathtt{a}}*\overline{\mathtt{c}}$ represent T_3. □

Theorem 12 *Let $T_n \in \{\mathtt{Lex}_n, \mathtt{Gray}_n\}$ be the lexicographic or the reflected Gray encoding of length n. Every interval $I = [x, y]$ can be represented by* $\max(n, 2n-4)$ *TCAM rules.*

Proof. We only show the proof for the lexicographic encoding; the proof for the Gray encoding is similar. For $n \leq 4$, it can be manually verified that every interval I can be represented by n or less TCAM rules. For $n \geq 5$, if the interval tree has $2n-2$ leaves, then by proof of Theorem 9 it can be represented by n TCAM rules. If the skeleton tree has $2n-3$ leaves, then

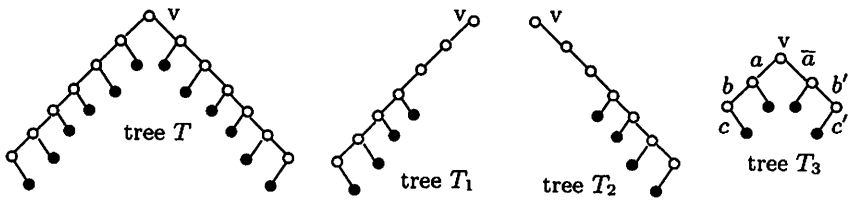

Figure 6: The case when skeleton tree has maximal number of prefix rules $2n-2$. T is decomposed into T_1, T_2, and T_3. For T_1 and T_2 we use $2n-6$ prefix rules. For T_3 three general rules are enough (instead of 4 prefix rules).

it has one of the formats shown in Figure 7 or the symmetric reflections of 7-(b) and 7-(c) around their roots, where C_l and C_r are chains. In all these three cases it can be proved that the leaf labeled by letter B can be represented by a combination of the TCAM rules that represent the other labeled leaves and thus (by Lemma 4) the whole interval can be represented by $2n-4$ TCAM rules. For instance, in the skeleton tree of Figure 7-(a), changing the second bit and the first bit of the prefix rules that represent the leaves A and D, respectively, to $*$ results in two rules that represent A' and D', either. As such, representing leaf B does not need a separate TCAM rule. □

6 Conclusion and future work

In this paper we studied the problem of interval representation in TCAM. This problem appears in the Internet routers that classify and filter the packets by implementing ACL policy rules.

The available methods of interval representation by TCAM use the lexicographic encoding to encode the integers of a given interval. We broadened this choice and explored the problem assuming that the underlying encoding scheme belongs to the family of dense-tree encoding schemes which includes the lexicographic encoding and the binary reflected Gray encoding.

We introduced the notion of a skeleton tree of a set $X \subseteq \{0,1\}^n$ and used it to devise a linear time algorithm which determines whether X is an interval in some dense tree encoding scheme.

Skeleton trees appear to be useful also in estimating the sizes of TCAMs. We also showed that the number of prime implicants (called here the *essential rules*) of a given interval set $X \subseteq \{0,1\}^n$ is at most $n(n-1)+1$ and $2n$ for the lexicographic encoding and the reflective Gray encoding,

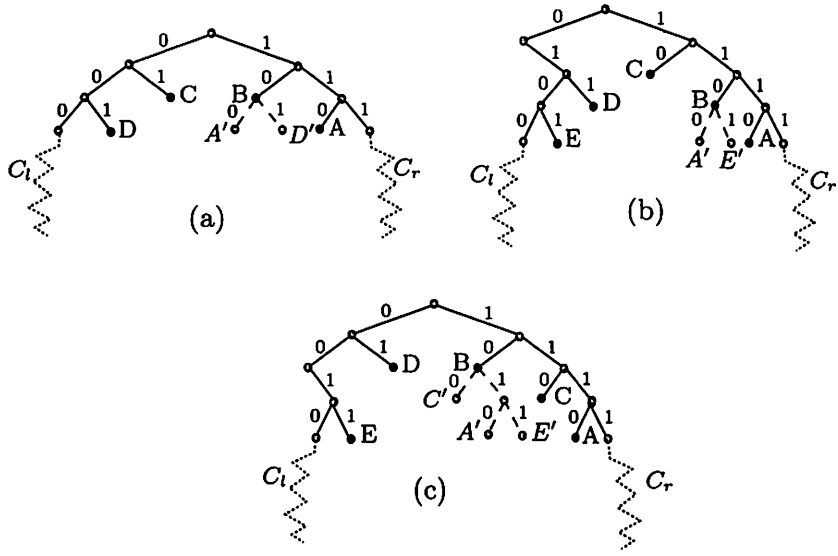

Figure 7: Three possible forms of a skeleton tree with $2n-3$ leaves, for $n \geq 5$.

respectively.

Finally, we proved that for the class of dense-tree encoding schemes of height n, every interval can be represented by $2n-3$ TCAM rules and in addition, there are intervals which need at least $\max(n, 2n-4)$ rules for their TCAM representation.

Suppose that we use a fixed $n-$bit encoding scheme for representing the integer values of a domain $D = [0, 2^n - 1]$. In the continuation of our work, we would devise an algorithm for finding a TCAM representation of a given interval over D with minimum number of TCAM rules. This problem is a special case of finding a TCAM representation of an arbitrary subset of the integers of domain D. The latter problem is an NP-hard problem as it can be polynomially reduced to the problem of the Minimal Disjunctive Normal Form of a Boolean formula [4] or to the problem of the Two-Level Logic Minimization [1].

References

[1] Robert K. Brayton, Gary D. Hachtel, Curtis T. McMullen, and Alberto L. Sangiovanni-Vincentelli. *Logic Minimization Algorithms for VLSI*

Synthesis. Kluwer Academic Publishers, 1984.

[2] G. Davis, C. Jeffries, and J. van Lunteren. Method and system for performing range rule testing in a ternary content addressable memory, April 2005. US Patent 6,886,073.

[3] E. N. Gilbert. Gray codes and paths on the n-cube. *Bell Systems Technical Journal*, 37:815–826, 1958.

[4] J. F. Gimpel. A method of producing a boolean function having an arbitrarily prescribed prime implicant table. *IEEE Trans. Computers*, 14:485–488, 1965.

[5] F. Gray. Pulse code communications, March 1953. US Patent 2,632,058.

[6] Hae-Jin Jeong, Il-Seop Song, Taeck-Geun Kwon, and Yoo-Kyoung Lee. A multi-dimension rule update in a tcam-based high-performance network security system. In *AINA '06: Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 2 (AINA'06)*, pages 62–66, Washington, DC, USA, 2006. IEEE Computer Society.

[7] R. Kempke and A. McAuley. Ternary cam memory architecture and methodology, August 1996. US Patent 5,841,874.

[8] T. Kohonen. *Content-Addressable Memories.* Springer-Verlag, New York, 1980.

[9] Karthik Lakshminarayanan, Anand Rangarajan, and Srinivasan Venkatachary. Algorithms for advanced packet classification with ternary cams. In *SIGCOMM*, pages 193–204, 2005.

[10] Huan Liu. Efficient mapping of range classifier into ternary-cam. In *HOTI '02: Proceedings of the 10th Symposium on High Performance Interconnects HOT Interconnects (HotI'02)*, page 95, Washington, DC, USA, 2002. IEEE Computer Society.

[11] V. Srinivasan, S. Suri, G. Varghese, and M. Waldvogel. Fast and scalable layer four switching. In *Proceedings of ACM Sigcomm*, volume 28, pages 191–202, October 1998.