ISSN 2313-1527 (PRINT)
ISSN 2313-1535 (ONLINE)

IJEIE

International Journal of Electronics and Information Engineering

Vol. 3, No. 2 (Dec. 2015)

INTERNATIONAL JOURNAL OF ELECTRONICS & INFORMATION ENGINEERING

PUBLISHING OFFICE

Min-Shiang Hwang
Department of Computer Science & Information Engineering, Asia University, Taichung 41354, Taiwan, R.O.C.
Email: mshwang@asia.edu.tw

International Journal of Electronics and Information Engineering is published both in traditional paper form (ISSN 2313-1527) and in Internet (ISSN 2313-1535) at http://ijeie.jalaxy.com.tw

PUBLISHER: Candy C. H. Lin

23-75, P.O. Box, Taichung, Taiwan 40199, R.O.C.

International Journal of Electronics and Information Engineering

Vol. 3, No. 2 (Dec. 1, 2015)

The Encryption Algorithms AES-PES16-1 and AES-RFWKPES16-1 Based on Networks PES16-1 and RFWKPES16-1

Tuychiev Gulom

Department Informatics and Applied Programming, National University of Uzbekistan
Student sity, 100174 Tashkent, Republic of Uzbekistan
(Email: blasterjon@gmail.com)
(Received Aug. 12, 2015; revised and accepted Sept. 23, 2015)

Abstract

In this article we developed a new block encryption algorithm based on networks PES16-1 and RFWKPES16-1 using of the transformations of the encryption algorithm AES, which is called AES-PES16-1 and AES-RFWKPES16-1. The block's length of this encryption algorithm is 256 bits, the number of rounds are 10, 12 and 14. The advantages of the encryption algorithms are that, when encryption and decryption process used the same algorithm. In addition, the encryption algorithm AES-PES16-1 encrypts faster than AES

Keywords: Advanced Encryption Standard; Feystel Network; Lai-Massey Scheme; Round Function; Round Keys; Output Transformation.

1 Introduction

In September 1997, the National Institute of Standards and Technology issued a public call for proposals for a new block cipher to succeed the Data Encryption Standard [28]. Out of 15 submitted algorithms the Rijndael cipher by Daemen and Rijmen [11] was chosen to become the new Advanced Encryption Standard in November 2001 [27]. The Advanced Encryption Standard is a block cipher with a fixed block length of 128 bits. It supports three different key lengths: 128 bits, 192 bits, and 256 bits. Encrypting a 128-bit block means transforming it in n rounds into a 128-bit output block. The number of rounds n depends on the key length: n =10 for 128-bit keys, n =12 for 192-bit keys, and n=14 for 256-bit keys. The 16-byte input block $(t_0, t_1, \cdots, t_{15})$ which is transformed during encryption is usually written as a 4x4 byte matrix, the called AES *State*.

t_0	t_4	t_8	t_{12}
t_1	t_5	t_9	t_{13}
t_2	t_6	t_{10}	t_{14}
t_3	t_7	t_{11}	t_{15}

The structure of each round of AES can be reduced to four basic transformations occurring to the elements of the *State*. Each round consists in applying successively to the *State* the SubBytes(), ShiftRows(), MixColumns() and AddRoundKey() transformations. The first round does the same with an extra AddRoundKey() at the beginning whereas the last round excludes the MixColumns() transformation.

The SubBytes() transformation is a nonlinear byte substitution that operates independently on each byte of the *State* using a substitution table (S-box). Figure 1 illustrates the SubBytes() transformation on the *State*.

In the ShiftRows() transformation operates on the rows of the *State*; it cyclically shifts the bytes in each row by a certain offset. For AES, the first row is left unchanged. Each byte of the second row is shifted one to the left.

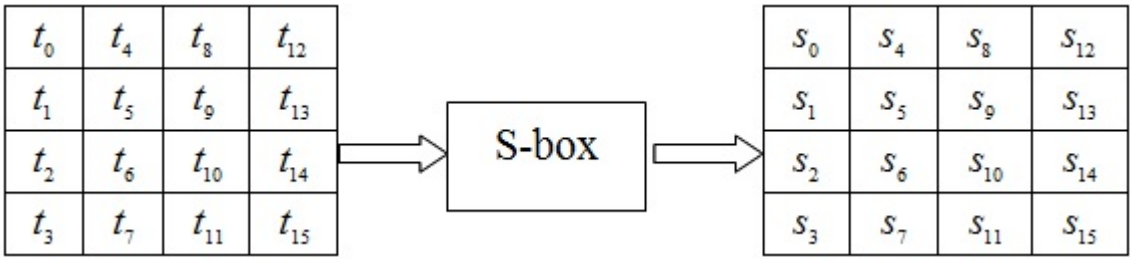

Figure 1: SubBytes() transformation

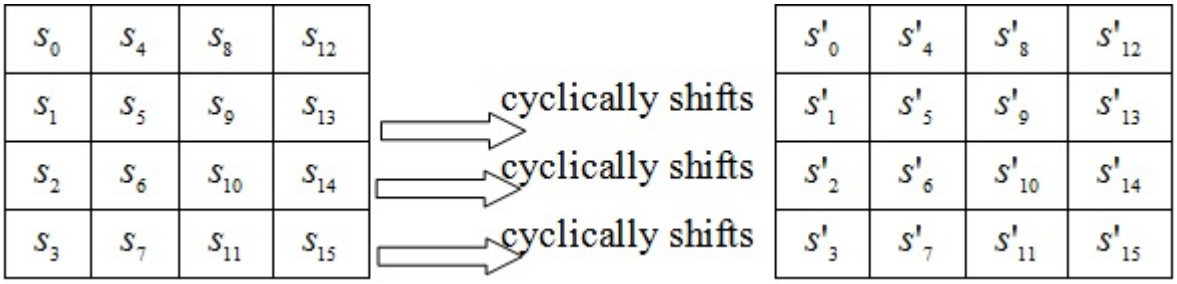

Figure 2: ShiftRows() transformation

Similarly, the third and fourth rows are shifted by offsets of two and three respectively. Figure 2 illustrates the ShiftRows() transformation.

The MixColumns() transformation operates on the *State* column-by-column, treating each column as a four-term polynomial. The columns are considered as polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial $a(x)$, given by $a(x) = 3x^2 + x^2 + x + 2$. Let $p = a(x) \otimes s'$:

$$\begin{bmatrix} p_{4i} \\ p_{4i+1} \\ p_{4i+2} \\ p_{4i+3} \end{bmatrix} = \begin{bmatrix} 02\ 03\ 01\ 01 \\ 01\ 02\ 03\ 01 \\ 01\ 01\ 02\ 03 \\ 03\ 01\ 01\ 02 \end{bmatrix} \begin{bmatrix} s'_{4i} \\ s'_{4i+1} \\ s'_{4i+2} \\ s'_{4i+3} \end{bmatrix}, \ i = \overline{0...3}$$

As a result of this multiplication, the four bytes in a column are replaced by the following:

$$y_{4i} = (\{02\} \bullet s'_{4i}) \oplus (\{03\} \bullet s'_{4i+1}) \oplus s'_{4i+2} \oplus s'_{4i+3}$$
$$y_{4i+1} = s'_{4i} \oplus (\{02\} \bullet s'_{4i+1}) \oplus (\{03\} \bullet s'_{4i+2}) \oplus s'_{4i+3}$$
$$y_{4i+2} = s'_{4i} \oplus s'_{4i+1} \oplus (\{02\} \bullet s'_{4i+2}) \oplus (\{03\} \bullet s'_{4i+3})$$
$$y_{4i+4} = (\{03\} \bullet s'_{4i}) \oplus s'_{4i+1} \oplus s'_{4i+2} \oplus (\{02\} \bullet s'_{4i+3}).$$

Figure 3 illustrates the MixColumns() transformation.

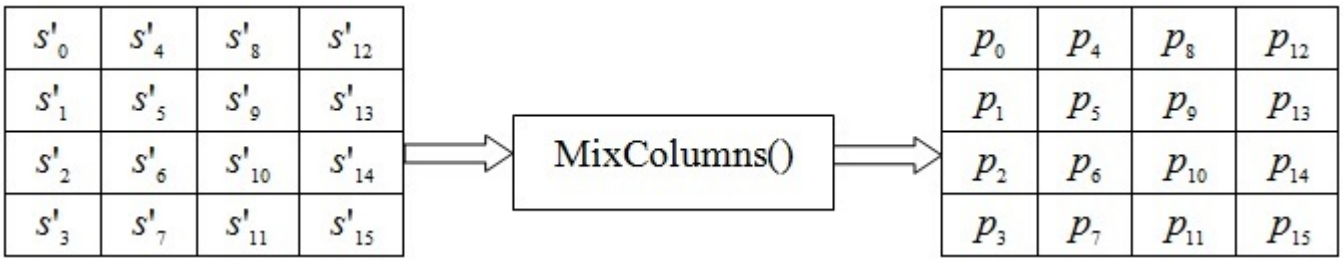

Figure 3: MixColumns() transformation

In the AddRoundKey() transformation, a round key is added to the *State* by a simple bitwise XOR operation. Figure 4 illustrates the AddRoundKey() transformation.

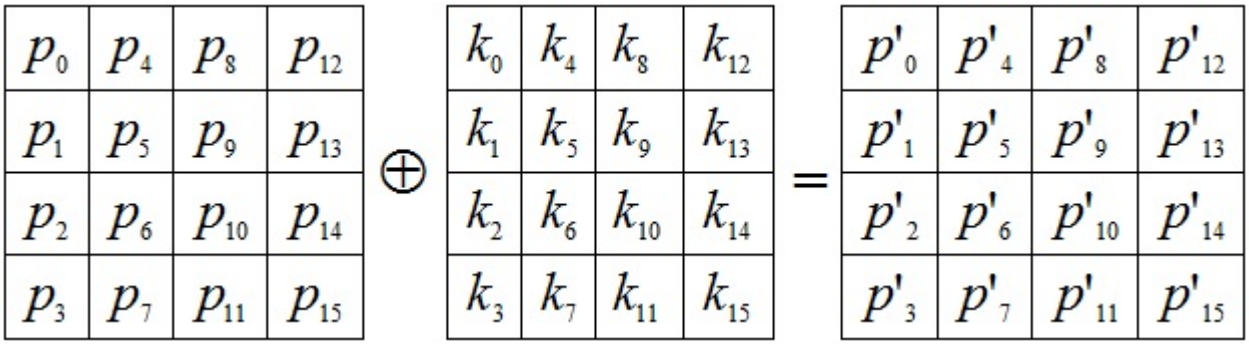

Figure 4: AddRoundKey() transformation

2 Analysis of AES, PES and IDEA

The first attack is a SQUARE attack suggested in [14] which uses 2^{128} - 2^{119} chosen plaintexts and 2^{120} encryptions. The second attack is a meet-in-the-middle attack proposed in [15] that requires 2^{32} chosen plaintexts and has a time complexity equivalent to almost 2128 encryptions. Recently, another attack on 7-round AES-128 was presented in [1]. The new attack is an impossible differential attack that requires $2^{117.5}$ chosen plaintexts and has a running time of 2^{121} encryptions. Similar results, but with better attack algorithms and lower complexities were reported in [41]. The resulting impossible differential attack on 7-round AES-192 has a data complexity of 292 chosen plaintexts and time complexity of 2^{162} encryptions, while the attack on AES-256 uses $2^{116.5}$ chosen plaintexts and running time of $2^{247.5}$ encryptions.

There are several attacks on AES-192 [1, 13, 14, 24, 29, 41]. The two most notable ones are the SQUARE attack on 8-round AES-192 presented in [14] that requires almost the entire code book and has a running time of 2^{188} encryptions and the meet in the middle attack on 7-round AES-192 in [13] that requires 2^{34+n} chosen plaintexts and has a running time of $2^{208-n} + 2^{82+n}$ encryptions. Legitimate values for n in the meet in the middle attack on AES-192 are $94 < n < 17$, thus, the minimal data complexity is 2^{51} chosen plaintexts (with time complexity equivalent to exhaustive search), and the minimal time complexity is 2^{146} (with data complexity of 2^{97} chosen plaintexts). AES-256 is analyzed in [1, 13, 14, 24, 41]. The best attack is the meet in the middle attack in [13] which uses 2^{32} chosen plaintexts and has a total running time of 2^{209} encryptions. Finally, we would like to note the existence of many related-key attacks on AES-192 and AES-256. As the main issue of this paper is not related-key attacks, and as we deal with the single key model, we do not elaborate on the matter here, but the reader is referred to [42] for the latest results on related-key impossible differential attacks on AES and to [18] for the latest results on related-key rectangle attacks on AES.

The strength of AES with respect to impossible differentials was challenged several times. The first attack of this kind is a 5-round attack presented in [4]. This attack is improved in [10] to a 6-round attack. In [29], an impossible differential attack on 7-round AES-192 and AES-256 is presented. The latter attack uses 2^{92} chosen plaintexts (or $2^{92.5}$ chosen plaintexts for AES-256) and has a running time of 2^{186} encryptions (or $2^{250.5}$ encryptions for AES-256). The time complexity of the latter attack was improved in [41] to 2^{162} encryptions for AES-192. In [1] a new 7-round impossible differential attack was presented. The new attack uses a different impossible differential, which is of the same general type as the one used in previous attacks (but has a slightly different structure). Using the new impossible differential leads to an attack that requires $2^{117.5}$ chosen plaintexts and has a running time of 2^{121} encryptions. This attack was later improved in [2, 41] to use $2^{115.5}$ chosen plaintexts with time complexity of 2^{119} encryptions.

The last application of impossible differential cryptanalysis to AES was the extension of the 7-round attack from [1] to 8-round AES-256 in [41]. The extended attack has a data complexity of $2^{116.5}$ chosen plaintexts and time complexity of $2^{247.5}$ encryption. We note that there were three more claimed impossible differential attacks on AES in [7, 8, 9]. However, as all these attacks are flawed. In paper [23] present a new attack on 7-round AES-128, a new attack on 7-round AES-192, and two attacks on 8-round AES-256. The attacks are based on the attacks proposed in [1, 29] but use additional techniques, including the early abort technique and key schedule considerations.

The best attack we present on 8-round AES-256 requires $2^{89.1}$ chosen plaintexts and has a time complexity of $2^{129.7}$ memory accesses. These results are significantly better than any previously published impossible differential attack on AES. We summarize results along with previously known results in Table 1.

The Proposed Encryption Standard (PES) is a 64-bit block cipher, using a 128-bit key, designed by Lai and Massey in 1990 (see [21]) and was a predecessor to IDEA (International Data Encryption Algorithm) [20]. IDEA was

Table 1: A Summary of the Attacks on AES

Number of rounds	complexity		Attack type
	Data (CP)	Time	
AES-128			
7	2^{128} - 2^{119}	2^{120}	Square [14]
7	$2^{117.5}$	2^{121}	Impossible Differential [14]
7	$2^{117.5}$	2^{119}	Impossible Differential [2, 41]
7	2^{32}	2^{128}	Meet in the middle [15]
7	$2^{112.2}$	$2^{117.2}$ MA	Impossible Differential [23]
AES-192			
7	2^{32}	2^{184}	Square [24]
7	$19 \cdot 2^{32}$	2^{155}	Square [14]
7	2^{92}	$2^{186.2}$	Impossible Differential [29]
7	$2^{115.5}$	2^{119}	Impossible Differential [41]
7	2^{92}	2^{162}	Impossible Differential [41]
7	2^{34+n}	$2^{208-n} + 2^{82+n}$	Meet in the middle [13]
8	$2^{128} - 2^{119}$	2^{188}	Square [14]
7	$2^{113.8}$	$2^{118.8}$ MA	Impossible Differential [23]
7	$2^{91.2}$	$2^{139.2}$	Impossible Differential [23]
AES-256			
7	2^{32}	2^{200}	Square [24]
7	$21 \cdot 2^{32}$	2^{172}	Square [14]
7	$2^{92.5}$	$2^{250.5}$	Impossible Differential [29]
7	2^{32}	2^{208}	Meet in the middle [13]
7	2^{34+n}	$2^{208-n} + 2^{82+n}$	Meet in the middle [13]
7	$2^{115.5}$	2^{119}	Impossible Differential [41]
8	$2^{116.5}$	$2^{247.5}$	Impossible Differential [41]
8	$2^{128} - 2^{119}$	2^{204}	Square [14]
8	2^{32}	2^{209}	Meet in the middle [13]
7	$2^{113.8}$	$2^{118.8}$ MA	Impossible Differential [23]
7	2^{92}	2^{163} MA	Impossible Differential [23]
8	$2^{111.1}$	$2^{227.8}$ MA	Impossible Differential [23]
8	$2^{89.1}$	$2^{229.7}$ MA	Impossible Differential [23]

originally called IPES (Improved PES). PES iterates eight rounds plus an output transformation. The cryptanalysis of PES and IDEA presented on Table 2 and Table 3.

Table 2: A Summary of the Attacks on IDEA

Attack Type	Year	Attacked Rounds	Key Bits round	Chosen Plaintext	Time
Differential [25]	1993	2	32	2^{10}	2^{42}
Differential [12]	1993	2.5	32	2^{10}	2^{32}
Differential [25]	1993	2.5	96	2^{10}	2^{106}
Related-Key Differential [17]	1996	3	32	6	$6 \cdot 2^{32}$
Differential-Linear [6]	1996	3	32	2^{30}	2^{44}
Differential [5]	1996	3	32	2^{30}	$0.75 \cdot 2^{44}$
Truncated Differential [19, 6]	1997	3.5	48	2^{56}	2^{67}
Miss-in-the-middle [3]	1998	3.5	64	$2^{38.5}$	2^{53}
Miss-in-the-middle [3]	1998	4	69	2^{37}	2^{70}
Related-Key Differential-Linear [16]	1998	4	15	38.3	-
Miss-in-the-Middle [3]	1998	4.5	80	2^{64}	2^{112}
Square attack [26]	2000	2.5	77	$3 \cdot 2^{16}$	$2^{63} + 2^{47}$
Square attack [26]	2000	2.5	31	2^{32}	2^{62}
Square [26]	2000	2.5	31	2^{48}	2^{79}
Related-Key Square [26]	2001	2.5	32	2	2^{41}

Table 3: A Summary of the Attacks on PES

Attack Type	Year	Attacked Rounds	Key Bits round	Chosen Plaintext	Time
Differential [22]	1991	7	96	2^{64}	2^{160}
Square [26]	2000	2.5	31	2^{17}	2^{47}
Square [26]	2001	2.5	31	2^{32}	2^{63}
Related-Key Square [26]	2001	2.5	32	2	241

On the basis of encryption algorithm PES and scheme Lai-Massey developed the networks PES16-1 and RFWKPES16-1, consisting from one round function [30, 36]. In the networks PES16-1 and RFWKPES16-1, similarly as in the Feistel network, when it encryption and decryption using the same algorithm. In the networks used one round function having four input and output blocks and as the round function can use any transformation.

Using transformation SubBytes(), ShiftRows(), MixColumns(), AddRoundKey() AES encryption algorithm as a round function networks IDEA8-1 [32], RFWKIDEA8-1 [32], PES8-1 [33], RFWKPES8-1 [34], IDEA16-1 [31], created encryption algorithms AES-IDEA8-1 [38], AES-RFWKIDEA8-1 [40], AES-PES8-1 [39], AES-RFWKPES8-1 [35], AES-IDEA16-1 [37].

In this paper developed block encryption algorithms AES-PES16-1 and AES-RFWKPES16-1 based networks PES16-1 [36] and RFWKPES16-1 [30] using transformation of the encryption algorithm AES. The length of block of the encryption algorithms is 256 bits, the number of rounds n equal to 10, 12, 14 and the length of key is variable from 256 bits to 1024 bits in steps 128 bits, i.e., key length is equal to 256, 384, 512, 640, 768, 896 and 1024 bits.

3 The Encryption Algorithm AES-PES16-1

3.1 The Structure of the Encryption Algorithm AES-PES16-1

In the encryption algorithm AES-PES16-1 as the round function used SubBytes(), ShiftRows(), MixColumns(), AddRoundKey() transformation encryption algorithm AES. The scheme n-rounded encryption algorithm AES-PES16-1 shown in Figure 5, and the length of subblocks X^0, X^1, $\cdots$, X^{15}, length of round keys $K_{17(i-1)}$, $K_{17(i-1)+1}$, $\cdots$,

$K_{17(i-1)+15}$, $i = \overline{1...n+1}$ and K_{17n+16}, K_{17n+17}, $\cdots$, K_{17n+47} are equal to 16-bits. A length of round key $K_{17(i-1)+16}$, $i = \overline{1...n}$ is 128-bits.

Consider the round function of the encryption algorithm AES-PES16-1. Initially 16-bit subblocks T^0, T^1, $\cdots$, T^7, are partitioned into 8-bit subblocks, i.e., on bytes: $t_0 = sb_0(T^0)$, $t_1 = sb_1(T^0)$, $t_2 = sb_0(T^1)$, $t_3 = sb_1(T^1)$, $t_4 = sb_0(T^2)$, $t_5 = sb_1(T^2)$, $t_6 = sb_0(T^3)$, $t_7 = sb_1(T^3)$, $t_8 = sb_0(T^4)$, $t_9 = sb_1(T^4)$, $t_{10} = sb_0(T^5)$, $t_{11} = sb_1(T^5)$, $t_{12} = sb_0(T^6)$, $t_{13} = sb_1(T^6)$, $t_{14} = sb_0(T^7)$, $t_{15} = sb_1(T^7)$, here $sb_0(X) = x_0x_1\cdots,x_7$, $sb_1(X) = x_8x_9\cdots,x_{15}$ and $X = x_0x_1\cdots,x_{15}$. After which the 8-bit subblocks t_0, t_1, $\cdots$, t_{15} are written into the *State* array and are executed the above transformations SubBytes(), ShiftRows(), MixColumns(), AddRoundKey(). In AddRoundKey() transformation as the key k_0, k_1, $\cdots$, k_{15} selected 128-bit round key $K_{17(i-1)+16}$. Initially the key $K_{17(i-1)+16}$ is partitioned into 32-bit keys $K^0_{17(i-1)+16}$, $K^1_{17(i-1)+16}$, ..., $K^7_{17(i-1)+16}$. These keys are partitioned into 8-bit subblocks as follows: $k_0 = sb_0(K^0_{17(i-1)+16})$, $k_1 = sb_1(K^0_{17(i-1)+16})$, $k_2 = sb_0(K^1_{17(i-1)+16})$, $k_3 = sb_1(K^1_{17(i-1)+16})$, $k_4 = sb_0(K^2_{17(i-1)+16})$, $k_5 = sb_1(K^2_{17(i-1)+16})$, $k_6 = sb_0(K^3_{17(i-1)+16})$, $k_7 = sb_1(K^3_{17(i-1)+16})$, $k_8 = sb_0(K^4_{17(i-1)+16})$, $k_9 = sb_1(K^4_{17(i-1)+16})$, $k_{10} = sb_0(K^5_{17(i-1)+16})$, $k_{11} = sb_1(K^5_{17(i-1)+16})$, $k_{12} = sb_0(K^6_{17(i-1)+16})$, $k_{13} = sb_1(K^6_{17(i-1)+16})$, $k_{14} = sb_0(K^7_{17(i-1)+16})$, $k_{15} = sb_1(K^7_{17(i-1)+16})$.

After the AddRoundKey() transformation we obtain 8-bits subblocks p'_0, p'_1, ..., p'_{15}. The resulting 8-bit subblocks are writes on a 16-bit subblocks Y^0, Y^1, $\cdots$, Y^7 as follows: $Y^0 = p'_0||p'_1$, $Y^1 = p'_2||p'_3$, $\cdots$, $Y^7 = p'_{14}||p'_{15}$.

The S-box SubBytes() transformation shown in Table 4 and is the only nonlinear transformation. The length of the input and output blocks S-box is eight bits. For example, if the input value the S-box is equal to 0xE7, then the output value is equal 0xE1, i.e. selected elements of intersection row 0xE and column 0x7.

Table 4: The S-box of encryption algorithm AES-PES16-1

	0x0	**0x1**	**0x2**	**0x3**	**0x4**	**0x5**	**0x6**	**0x7**	**0x8**	**0x9**	**0xA**	**0xB**	**0xC**	**0xD**	**0xE**	**0xF**
0x0	0x01	0x2B	0xB0	0x91	0x87	0x1E	0x20	0x75	0xE9	0x59	0x3F	0xFD	0xA6	0x86	0x0C	0x48
0x1	0xD0	0x2C	0x54	0xD3	0x98	0x2A	0xF8	0xFB	0x4E	0x02	0xEB	0x32	0xFE	0xA9	0x76	0xB3
0x2	0xA2	0x22	0x5B	0x24	0xAB	0x4C	0x41	0x92	0xD7	0x51	0xB2	0x03	0xF2	0x23	0x11	0xE8
0x3	0x9F	0x09	0xE2	0x37	0x31	0x9D	0x67	0xBC	0x1B	0x52	0x50	0x08	0xEF	0xDC	0x64	0xCC
0x4	0x46	0xD4	0x47	0x4D	0xB5	0xD8	0xAE	0x97	0xB1	0x72	0x7E	0x9C	0x81	0xF0	0xC3	0x6E
0x5	0x49	0xC4	0xA1	0xAD	0x66	0xDA	0xE0	0x56	0xE6	0x28	0x45	0x0E	0x21	0x63	0xD2	0x16
0x6	0x3C	0xD1	0xF4	0x60	0xC6	0x9B	0x6D	0x96	0x84	0x1C	0xDD	0xAF	0xCD	0xC8	0x7A	0x88
0x7	0x35	0x13	0x42	0xF5	0xA3	0x06	0xF6	0x3D	0x39	0x5A	0x5F	0x9E	0x2E	0x5E	0x7F	0x40
0x8	0x6A	0x80	0xAA	0x55	0x68	0x65	0x7C	0x94	0x8D	0x2D	0x57	0x30	0xBB	0x14	0xC9	0x8E
0x9	0x85	0xA8	0xE7	0x1D	0x1A	0x05	0xDF	0x4F	0x00	0x29	0x07	0x10	0x61	0x90	0x3A	0xA7
0xA	0x74	0x95	0x8A	0xC2	0xA5	0xE3	0x58	0xEC	0xCF	0xB8	0xB6	0xF9	0x27	0x19	0x4A	0xED
0xB	0xCE	0xAC	0x53	0x5C	0x89	0xFA	0x1F	0x70	0xA4	0x8F	0xC5	0x17	0x43	0x69	0xCA	0xBE
0xC	0x7B	0x15	0xA0	0x3B	0x0F	0xD5	0x26	0xF1	0x6B	0xE5	0x34	0x8B	0xD9	0xC1	0xCB	0x33
0xD	0x0A	0xB4	0xDE	0xC7	0x5D	0x25	0xB9	0xFC	0x7D	0x2F	0x77	0xFF	0x12	0x4B	0xF7	0x82
0xE	0x8C	0xE4	0xC0	0xD6	0x62	0xDB	0x0D	0xE1	0x44	0x73	0xEA	0x93	0xEE	0x0B	0x79	0x99
0xF	0x71	0x6F	0xB7	0x78	0xBD	0x36	0x3E	0x9A	0xBA	0x38	0xBF	0x04	0x18	0xF3	0x83	0x6C

Consider the encryption process of encryption algorithm AES-PES16-1. Initially the 256-bit plaintext X partitioned into subblocks of 16-bits X^0_0, X^1_0, $\cdots$, X^{15}_0, and performs the following steps:

1) subblocks X^0_0, X^1_0, $\cdots$, X^{15}_0 summed by XOR respectively with round key K_{17n+8}, K_{17n+16}, K_{17n+17}, ..., K_{17n+31}: $X^j_0 = X^j_0 \oplus K_{17n+16+j}$, $j = \overline{0...15}$.

2) subblocks X^0_0, X^1_0, $\cdots$, X^{15}_0 multiplied and summed respectively with the round keys $K_{17(i-1)}$, $K_{17(i-1)+1}$, $\cdots$, $K_{17(i-1)+7}$ and calculated 16-bit subblocks

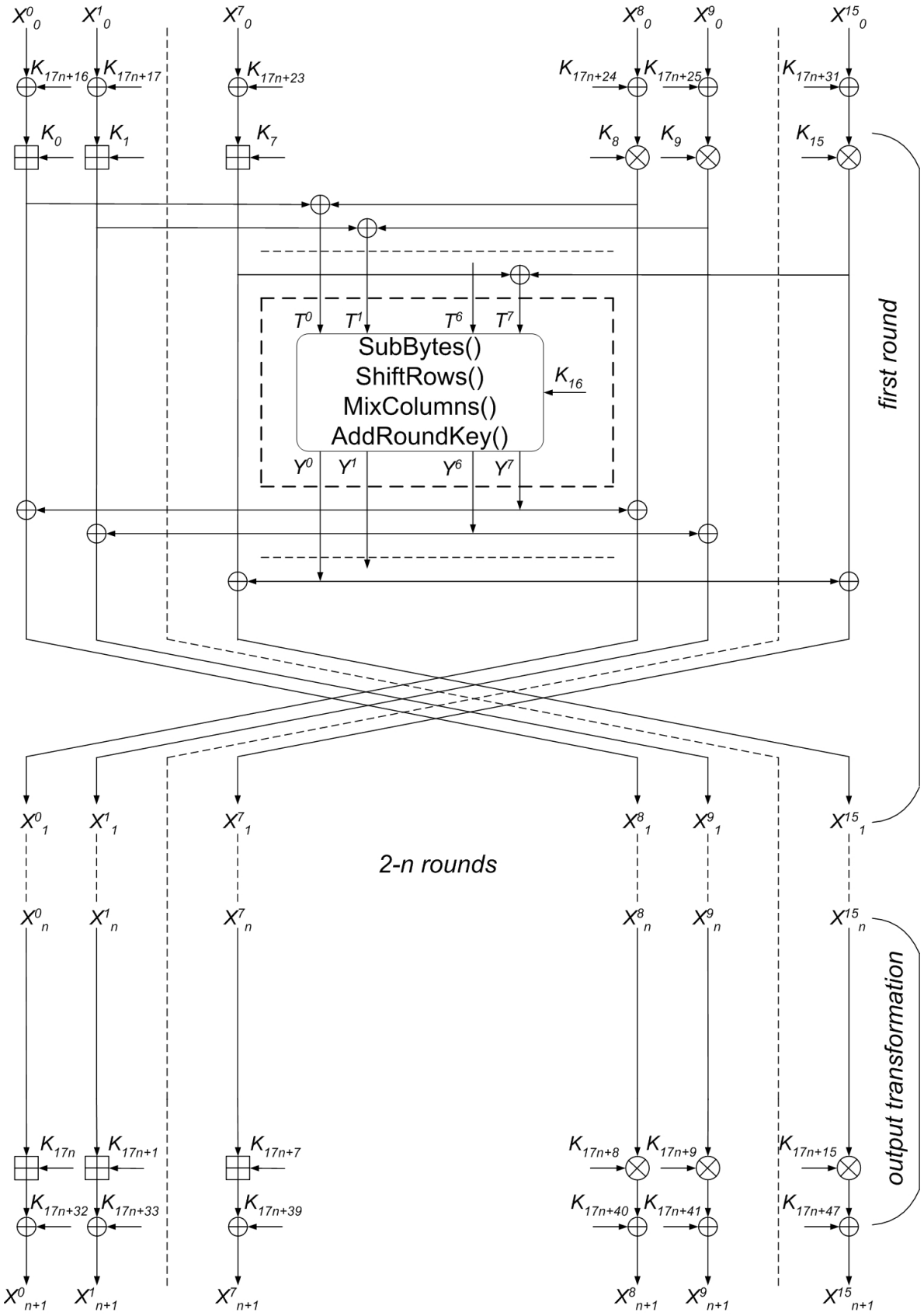

Figure 5: The scheme n-rounded encryption algorithm AES-PES16-1

T^0, T^1, ..., T^7. This step can be represented as follows:

$$\begin{aligned}
T^0 &= (X^0_{i-1} + K_{17(i-1)}) \oplus (X^8_{i-1} \cdot K_{17(i-1)+8}, \\
T^1 &= (X^1_{i-1} + K_{17(i-1)+1}) \oplus (X^9_{i-1} \cdot K_{17(i-1)+9}), \\
T^2 &= (X^2_{i-1} + K_{17(i-1)+2}) \oplus (X^{10}_{i-1} \cdot K_{17(i-1)+10}), \\
T^3 &= (X^3_{i-1} + K_{17(i-1)+3}) \oplus (X^{11}_{i-1} \cdot K_{17(i-1)+11}), \\
T^4 &= (X^4_{i-1} + K_{17(i-1)+4}) \oplus (X^{12}_{i-1} \cdot K_{17(i-1)+12}), \\
T^5 &= (X^5_{i-1} + K_{17(i-1)+5}) \oplus (X^{13}_{i-1} \cdot K_{17(i-1)+13}), \\
T^6 &= (X^6_{i-1} + K_{17(i-1)+6}) \oplus (X^{14}_{i-1} \cdot K_{17(i-1)+14}), \\
T_7 &= (X^7_{i-1} + K_{17(i-1)+7}) \oplus (X^{15}_{i-1} \cdot K_{17(i-1)+15}), i = 1.
\end{aligned}$$

3) subblocks T^0, T^1, $\cdots$, T^7 is split into 8-bit subblocks t_0, t_1, $\cdots$, t_{15} and performed SubBytes(), ShiftRows(), MixColumns(), AddRoundKey() transformation. Output subblocks of the round function of the encryption algorithm are Y^0, Y^1, ..., Y^7.

4) subblocks Y^0, Y^1, $\cdots$, Y^7 are summed to XOR with subblocks X^0_{i-1}, X^1_{i-1}, ..., X^8_{i-1}, i.. $X^j_{i-1} = X^j_{i-1} \oplus Y_{7-j}$, $X^{j+8}_{i-1} = X^{j+8}_{i-1} \oplus Y_{7-j}$, $j = \overline{0...7}$, $i = 1$.

5) at the end of the round subblocks X^j_{i-1} and X^{j+8}_{i-1}, $j = \overline{0...7}$ swapped, i.., $X^j_i = X^{j+8}_{i-1}$, $X^{j+8}_i = X^j_{i-1}$, $j = \overline{0...7}$, $i = 1$.

6) repeating steps 2-5 n times, i.e., $i = \overline{2...n}$ obtain subblocks X^0_n, X^1_n, $\cdots$, X^{15}_n.

7) in output transformation round keys are multiplied and summed into subblocks, i.e. $X^j_{n+1} = X^j_n + K_{17n+j}$, $X^{8+j}_{n+1} = X^{8+j}_n \cdot K_{17n+8+j}$, $j = \overline{0...7}$.

8) subblocks X^0_{n+1}, X^1_{n+1}, $\cdots$, X^{15}_{n+1} are summed to XOR with the round key K_{17n+32}, K_{17n+33}, $\cdots$, K_{17n+47}: $X^j_{n+1} = X^j_{n+1} \oplus K_{17n+32+j}$, $j = \overline{0...15}$. As ciphertext plaintext X receives the combined 16-bit subblocks $X^0_{n+1}||X^1_{n+1}||...||X^{15}_{n+1}$.

3.2 Key Generation of the Encryption Algorithm AES-PES16-1

In n-round encryption algorithm AES-PES16-1 in each round we applied sixteen round keys of the 16-bit, one round keys of the 128-bit and output transformation sixteen round keys of the 16-bit. In addition, before the first round and after the output transformation we used sixteen round keys of 16-bits. Total number of 16-bit round keys is equal to $16n$+48, and 128-bit round key is equal to n. If 128-bit round key convert to eight 16-bit keys, the total number of 16-bit keys equal to $24n$+48. In Figure 5 encryption used encryption round keys K^c_i instead of K_i, while decryption used decryption round keys K^d_i.

If n=10 then need 288 to generate round keys, if n=12, you need to generate 336 round keys and if n=14 need 384 to generate round keys.

When generating round keys like the AES encryption algorithm uses an array Rcon: Rcon=[0x0001, 0x0002, 0x0004, 0x0008, 0x0010, 0x0020, 0x0040, 0x0080, 0x0100, 0x0200, 0x0400, 0x0800, 0x1000, 0x2000, 0x4000, 0x8000].

The key encryption algorithm K of length l ($256 \le l \le 1024$) bits is divided into 16-bit round keys $K^{c'}_0$, $K^{c'}_1$,..., $K^{c'}_{Lenght-1}$, $Lenght = l/16$, here $K = \{k_0, k_1, ..., k_{l-1}\}$, $K^{c'}_0 = \{k_0, k_1, ..., k_{15}\}$, $K^{c'}_1 = \{k_{16}, k_9, ..., k_{31}\}$,..., $K^{c'}_{Lenght-1} = \{k_{l-16}, k_{l-15}, ..., k_{l-1}\}$ and $K = K^{c'}_0 || K^{c'}_1 ||...||K^{c'}_{Lenght-1}$. Then we calculate $K_L = K^{c'}_0 \oplus K^{c'}_1 \oplus ... \oplus K^{c'}_{Lenght-1}$. If $K_L = 0$ then K_L is chosen as 0xC5C3, i.e. K_L = 0xC5C3. When generating a round keys $K^{c'}_i$, $i = \overline{Lenght...24n+47}$, we used transformation *SubBytes16()* and *RotWord16()*, here *SubBytes16()*-is transformation 16-bit subblock into S-box and $SubBytes16(X) = S(sb_0(X))||S(sb_1(X)))$, *RotWord16()*-cyclic shift to the left of 1 bit of the 16-bit subblock. When the condition $imod3 = 1$ is true, then the round keys are computed as $K^{c'}_i = SubBytes16(K^{c'}_{i-Lenght+1}) \oplus SubBytes16(RotWord16(K^{c'}_{i-Lenght})) \oplus Rcon[imod16] \oplus K_L$, otherwise $K^{c'}_i = SubBytes16(K^{c'}_{i-Lenght}) \oplus SubBytes16(K^{c'}_{i-Lenght+1}) \oplus K_L$. After each round key generation the value K_L is cyclic shift to the left by 1 bit.

Decryption round keys are computed on the basis of encryption round keys and decryption round keys of the output transformation associate with of encryption round keys as follows:

$$\begin{aligned}&(K_{24n}^{d'}, K_{24n+1}^{d'}, K_{24n+2}^{d'}, K_{24n+3}^{d'}, K_{24n+4}^{d'}, K_{24n+5}^{d'}, K_{24n+6}^{d'}, K_{24n+7}^{d'}, K_{24n+8}^{d'}, K_{24n+9}^{d'}, K_{24n+10}^{d'},\\ &\quad K_{24n+11}^{d'}, K_{24n+12}^{d'}, K_{24n+13}^{d'}, K_{24n+14}^{d'}, K_{24n+15}^{d'})\\ =\ &(-K_0^{c'}, -K_1^{c'}, -K_2^{c'}, -K_3^{c'}, -K_4^{c'}, -K_5^{c'}, -K_6^{c'}, -K_7^{c'}, (K_8^{c'})^{-1}, (K_9^{c'})^{-1}, (K_{10}^{c'})^{-1},\\ &\quad (K_{11}^{c'})^{-1}, (K_{12}^{c'})^{-1}, (K_{13}^{c'})^{-1}, (K_{14}^{c'})^{-1}, (K_{15}^{c'})^{-1}).\end{aligned}$$

For example, if the number of rounds is 10 the formula is as follows:

$$\begin{aligned}&(K_{240}^{d'}, K_{241}^{d'}, K_{242}^{d'}, K_{243}^{d'}, K_{244}^{d'}, K_{245}^{d'}, K_{246}^{d'}, K_{247}^{d'}, K_{248}^{d'}, K_{249}^{d'}, K_{250}^{d'}, K_{251}^{d'}, K_{252}^{d'}, K_{253}^{d'}, K_{254}^{d'}, K_{255}^{d'})\\ =\ &(-K_0^{c'}, -K_1^{c'}, -K_2^{c'}, -K_3^{c'}, -K_4^{c'}, -K_5^{c'}, -K_6^{c'}, -K_7^{c'}, (K_8^{c'})^{-1}, (K_9^{c'})^{-1}, (K_{10}^{c'})^{-1},\\ &\quad (K_{11}^{c'})^{-1}, (K_{12}^{c'})^{-1}, (K_{13}^{c'})^{-1}, (K_{14}^{c'})^{-1}, (K_{15}^{c'})^{-1}).\end{aligned}$$

Likewise, the decryption round keys of the first, second, third, and n–round associates with the encryption round keys as follows:

$$\begin{aligned}&(K_{24(i-1)}^{d'}, K_{24(i-1)+1}^{d'}, K_{24(i-1)+2}^{d'}, K_{24(i-1)+3}^{d'}, K_{24(i-1)+4}^{d'}, K_{24(i-1)+5}^{d'}, K_{24(i-1)+6}^{d'}, K_{24(i-1)+7}^{d'},\\ &\quad K_{24(i-1)+8}^{d'}, K_{24(i-1)+9}^{d'}, K_{24(i-1)+10}^{d'}, K_{24(i-1)+11}^{d'}, K_{24(i-1)+12}^{d'}, K_{24(i-1)+13}^{d'}, K_{24(i-1)+14}^{d'},\\ &\quad K_{24(i-1)+15}^{d'})\\ =\ &(-K_{24(n-i+1)}^{c'}, -K_{24(n-i+1)+1}^{c'}, -K_{24(n-i+1)+2}^{c'}, -K_{24(n-i+1)+3}^{c'}, -K_{24(n-i+1)+4}^{c'}, -K_{24(n-i+1)+5}^{c'},\\ &\quad -K_{24(n-i+1)+6}^{c'}, -K_{24(n-i+1)+7}^{c'}, (K_{24(n-i+1)+8}^{c'})^{-1}, (K_{24(n-i+1)+9}^{c'})^{-1}, (K_{24(n-i+1)+10}^{c'})^{-1},\\ &\quad (K_{24(n-i+1)+11}^{c'})^{-1}, (K_{24(n-i+1)+12}^{c'})^{-1}, (K_{24(n-i+1)+13}^{c'})^{-1}, (K_{24(n-i+1)+14}^{c'})^{-1},\\ &\quad (K_{24(n-i+1)+15}^{c'})^{-1}), i = \overline{1...n}\\ K_{24(i-1)+16+j}^{d'} \ =\ &K_{24(n-i+1)+16+j}^{c'}, j = \overline{0...15}, i = \overline{1...n}.\end{aligned}$$

If the number of rounds n is 10 the first decryption round keys associates with the encryption round keys as follows:

$$\begin{aligned}&(K_0^{d'}, K_1^{d'}, K_2^{d'}, K_3^{d'}, K_4^{d'}, K_5^{d'}, K_6^{d'}, K_7^{d'}, K_8^{d'}, K_9^{d'}, K_{10}^{d'}, K_{11}^{d'}, K_{12}^{d'}, K_{13}^{d'}, K_{14}^{d'}, K_{15}^{d'})\\ =\ &(-K_{240}^{c'}, -K_{241}^{c'}, -K_{242}^{c'}, -K_{243}^{c'}, -K_{244}^{c'}, -K_{245}^{c'}, -K_{246}^{c'}, -K_{247}^{c'}, (K_{248}^{c'})^{-1}, (K_{249}^{c'})^{-1}, (K_{250}^{c'})^{-1},\\ &\quad (K_{251}^{c'})^{-1}, (K_{252}^{c'})^{-1}, (K_{253}^{c'})^{-1}, (K_{254}^{c'})^{-1}, (K_{255}^{c'})^{-1})\\ K_{16+j}^{d'} \ =\ &K_{240+j}^{c'}, j = \overline{0...15}.\end{aligned}$$

Decryption round keys applied to the first round and after the output transformation associated with the encryption round keys as follows: $K_{24n+16+j}^{d'} = K_{24n+32+j}^{c'}$, $K_{24n+32+j}^{d'} = K_{24n+16+j}^{c'}$, $j = \overline{0...15}$.

Encryption round keys K_i^c associated with $K_i^{c'}$ follows: $K_{17i+j}^{c} = K_{24i+j}^{c'}$, $j = \overline{0...15}$, $K_{17i+16}^{c} = K_{24i+16}^{c'}||K_{24i+17}^{c'}||...||K_{24i+23}^{c'}$, $K_{17n+j}^{c} = K_{12n+j}^{c'}$, $j = \overline{0...15}$, $K_{17n+16+j}^{c} = K_{24n+16+j}^{c'}$, $j = \overline{0...31}$. Likewise, decryption round keys K_i^d associated with $K_i^{d'}$ follows: $K_{17i+j}^{d} = K_{24i+j}^{d'}$, $j = \overline{0...15}$, $K_{17i+16}^{d} = K_{24i+16}^{d'}||K_{24i+17}^{d'}||...||K_{24i+23}^{d'}$, $K_{17n+j}^{d} = K_{12n+j}^{d'}$, $j = \overline{0...15}$, $K_{17n+16+j}^{d} = K_{24n+16+j}^{d'}$, $j = \overline{0...31}$.

4 The Encryption Algorithm AES-RFWKPES16-1

4.1 The Structure of the Encryption Algorithm AES-RFWKPES16-1

In the encryption algorithm AES-RFWKPES16-1 as the round function used SubBytes(), ShiftRows(), MixColumns() transformation encryption algorithm AES. The scheme n-rounded encryption algorithm AES-RFWKPES16-1 shown in Figure 6, and the length of subblocks X^0, X^1, ..., X^{15}, length of round keys $K_{16(i-1)}$, $K_{16(i-1)+1}$, ..., $K_{16(i-1)+15}$,, $i = \overline{1...n+1}$ and K_{16n+16}, K_{16n+17}, ..., K_{16n+47} are equal to 16-bits.

Consider the round function of the encryption algorithm AES-RFWKPES16-1. Initially 16-bit subblocks T^0, T^1, $\cdots$, T^7, are partitioned into 8-bit subblocks, i.e., on bytes: $t_0 = sb_0(T^0)$, $t_1 = sb_1(T^0)$, $t_2 = sb_0(T^1)$, $t_3 = sb_1(T^1)$, $t_4 = sb_0(T^2)$, $t_5 = sb_1(T^2)$, $t_6 = sb_0(T^3)$, $t_7 = sb_1(T^3)$, $t_8 = sb_0(T^4)$, $t_9 = sb_1(T^4)$, $t_{10} = sb_0(T^5)$, $t_{11} = sb_1(T^5)$,

$t_{12} = sb_0(T^6)$, $t_{13} = sb_1(T^6)$, $t_{14} = sb_0(T^7)$, $t_{15} = sb_1(T^7)$. After which the 8-bit subblocks t_0, t_1, ..., t_{15} are written into the *State* array and are executed the above transformations SubBytes(), ShiftRows(), MixColumns().

After the MixColumns() transformation we obtain 8-bits subblocks p_0, p_1, ..., p_{15}. The resulting 8-bit subblocks are writes on a 16-bit subblocks Y^0, Y^1, $\cdots$, Y^7 as follows: $Y^0 = p_0||p_1$, $Y^1 = p_2||p_3$, $\cdots$, $Y^7 = p_{14}||p_{15}$.

The S-box SubBytes() transformation shown in Table 5 and is the only nonlinear transformation. The length of the input and output blocks S-box is eight bits.

Table 5: The S-box of encryption algorithm AES-RFWKPES16-1

	0x0	**0x1**	**0x2**	**0x3**	**0x4**	**0x5**	**0x6**	**0x7**	**0x8**	**0x9**	**0xA**	**0xB**	**0xC**	**0xD**	**0xE**	**0xF**
0x0	0x01	0x2B	0xB0	0x91	0x87	0x1E	0x20	0x75	0xE9	0x59	0x3F	0xFD	0xA6	0x86	0x0C	0x48
0x1	0xD0	0x2C	0x54	0xD3	0x98	0x2A	0xF8	0xFB	0x4E	0x02	0xEB	0x32	0xFE	0xA9	0x76	0xB3
0x2	0xA2	0x22	0x5B	0x24	0xAB	0x4C	0x41	0x92	0xD7	0x51	0xB2	0x03	0xF2	0x23	0x11	0xE8
0x3	0x9F	0x09	0xE2	0x37	0x31	0x9D	0x67	0xBC	0x1B	0x52	0x50	0x08	0xEF	0xDC	0x64	0xCC
0x4	0x46	0xD4	0x47	0x4D	0xB5	0xD8	0xAE	0x97	0xB1	0x72	0x7E	0x9C	0x81	0xF0	0xC3	0x6E
0x5	0x49	0xC4	0xA1	0xAD	0x66	0xDA	0xE0	0x56	0xE6	0x28	0x45	0x0E	0x21	0x63	0xD2	0x16
0x6	0x3C	0xD1	0xF4	0x60	0xC6	0x9B	0x6D	0x96	0x84	0x1C	0xDD	0xAF	0xCD	0xC8	0x7A	0x88
0x7	0x35	0x13	0x42	0xF5	0xA3	0x06	0xF6	0x3D	0x39	0x5A	0x5F	0x9E	0x2E	0x5E	0x7F	0x40
0x8	0x6A	0x80	0xAA	0x55	0x68	0x65	0x7C	0x94	0x8D	0x2D	0x57	0x30	0xBB	0x14	0xC9	0x8E
0x9	0x85	0xA8	0xE7	0x1D	0x1A	0x05	0xDF	0x4F	0x00	0x29	0x07	0x10	0x61	0x90	0x3A	0xA7
0xA	0x74	0x95	0x8A	0xC2	0xA5	0xE3	0x58	0xEC	0xCF	0xB8	0xB6	0xF9	0x27	0x19	0x4A	0xED
0xB	0xCE	0xAC	0x53	0x5C	0x89	0xFA	0x1F	0x70	0xA4	0x8F	0xC5	0x17	0x43	0x69	0xCA	0xBE
0xC	0x7B	0x15	0xA0	0x3B	0x0F	0xD5	0x26	0xF1	0x6B	0xE5	0x34	0x8B	0xD9	0xC1	0xCB	0x33
0xD	0x0A	0xB4	0xDE	0xC7	0x5D	0x25	0xB9	0xFC	0x7D	0x2F	0x77	0xFF	0x12	0x4B	0xF7	0x82
0xE	0x8C	0xE4	0xC0	0xD6	0x62	0xDB	0x0D	0xE1	0x44	0x73	0xEA	0x93	0xEE	0x0B	0x79	0x99
0xF	0x71	0x6F	0xB7	0x78	0xBD	0x36	0x3E	0x9A	0xBA	0x38	0xBF	0x04	0x18	0xF3	0x83	0x6C

Consider the encryption process of encryption algorithm AES-RFWKPES16-1. Initially the 256-bit plaintext X partitioned into subblocks of 16-bits X_0^0, X_0^1, $\cdots$, X_0^{15}, and performs the following steps:

1) subblocks X_0^0, X_0^1, $\cdots$, X_0^{15} summed by XOR respectively with round key K_{16n+16}, K_{16n+17}, $\cdots$, K_{16n+31}: $X_0^j = X_0^j \oplus K_{16n+16+j}$, $j = \overline{0...15}$.

2) subblocks X_0^0, X_0^1, $\cdots$, X_0^{15} multiplied and summed respectively with the round keys $K_{16(i-1)}$, $K_{16(i-1)+1}$, $\cdots$, $K_{16(i-1)+7}$ and calculated 16-bit subblocks T^0, T^1, ..., T^7. This step can be represented as follows:

$$\begin{aligned}
T^0 &= (X_{i-1}^0 + K_{16(i-1)}) \oplus (X_{i-1}^8 \cdot K_{16(i-1)+8}),\\
T^1 &= (X_{i-1}^1 + K_{16(i-1)+1}) \oplus (X_{i-1}^9 \cdot K_{16(i-1)+9}),\\
T^2 &= (X_{i-1}^2 + K_{16(i-1)+2}) \oplus (X_{i-1}^{10} \cdot K_{16(i-1)+10}),\\
T^3 &= (X_{i-1}^3 + K_{16(i-1)+3}) \oplus (X_{i-1}^{11} \cdot K_{16(i-1)+11}),\\
T^4 &= (X_{i-1}^4 + K_{16(i-1)+4}) \oplus (X_{i-1}^{12} \cdot K_{16(i-1)+12}),\\
T^5 &= (X_{i-1}^5 + K_{16(i-1)+5}) \oplus (X_{i-1}^{13} \cdot K_{16(i-1)+13}),\\
T^6 &= (X_{i-1}^6 + K_{16(i-1)+6}) \oplus (X_{i-1}^{14} \cdot K_{16(i-1)+14}),\\
T_7 &= (X_{i-1}^7 + K_{16(i-1)+7}) \oplus (X_{i-1}^{15} \cdot K_{16(i-1)+15}), i = 1.
\end{aligned}$$

3) subblocks T^0, T^1, $\cdots$, T^7 is split into 8-bit subblocks t_0, t_1, $\cdots$, t_{15} and performed SubBytes(), ShiftRows(), MixColumns() transformation. Output subblocks of the round function of the encryption algorithm are Y^0, Y^1, $\cdots$, Y^7.

4) subblocks Y^0, Y^1, $\cdots$, Y^7 are summed to XOR with subblocks X_{i-1}^0, X_{i-1}^1, ..., X_{i-1}^8, .. $X_{i-1}^j = X_{i-1}^j \oplus Y_{7-j}$, $X_{i-1}^{j+8} = X_{i-1}^{j+8} \oplus Y_{7-j}$, $j = \overline{0...7}$, $i = 1$.

5) at the end of the round subblocks X_{i-1}^j and X_{i-1}^{j+8}, $j = \overline{0...7}$ swapped, i.., $X_i^j = X_{i-1}^{j+8}$, $X_i^{j+8} = X_{i-1}^j$, $j = \overline{0...7}$, $i = 1$.

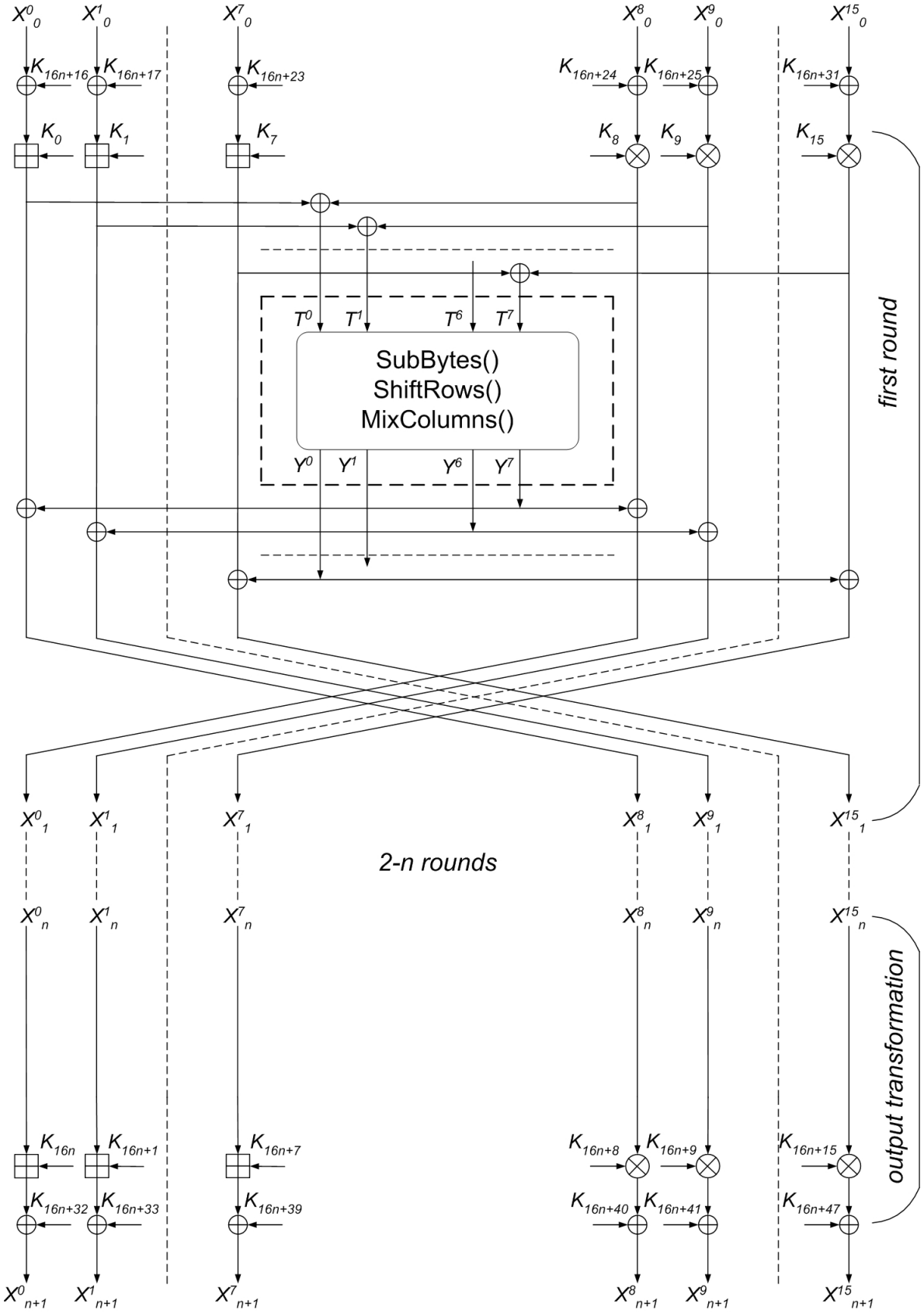

Figure 6: The scheme n-rounded encryption algorithm AES-RFWKPES16-1

6) repeating steps 2-5 n times, i.e., $i = \overline{2...n}$ obtain subblocks X_n^0, X_n^1, $\cdots$, X_n^{15}.

7) in output transformation round keys are multiplied and summed into subblocks, i.e. $X_{n+1}^j = X_n^j + K_{16n+j}$, $X_{n+1}^{8+j} = X_n^{8+j} \cdot K_{16n+8+j}$, $j = \overline{0...7}$.

8) subblocks X_{n+1}^0, X_{n+1}^1, $\cdots$, X_{n+1}^{15} are summed to XOR with the round key K_{16n+32}, K_{16n+33}, $\cdots$, K_{16n+47}: $X_{n+1}^j = X_{n+1}^j \oplus K_{166n+32+j}$, $j = \overline{0...15}$. As ciphertext plaintext X receives the combined 16-bit subblocks $X_{n+1}^0||X_{n+1}^1||...||X_{n+1}^{15}$.

4.2 Key Generation of the Encryption Algorithm AES-RFWKPES16-1

In n-round encryption algorithm AES-RFWKPES16-1 in each round we applied sixteen round keys of the 16-bit and output transformation sixteen round keys of the 16-bit. In addition, before the first round and after the output transformation we used sixteen round keys of 16-bits. Total number of 16-bit round keys is equal to $16n+48$. In Figure 6 encryption used encryption round keys K_i^c instead of K_i, while decryption used decryption round keys K_i^d.

The key encryption algorithm K of length l ($256 \le l \le 1024$)bits is divided into 16-bit round keys K_0^c, K_1^c,..., $K_{Lenght-1}^c$, $Lenght = l/16$, here $K = \{k_0, k_1, ..., k_{l-1}\}$, $K_0^c = \{k_0, k_1, ..., k_{15}\}$, $K_1^c = \{k_{16}, k_9, ..., k_{31}\}$,..., $K_{Lenght-1}^c = \{k_{l-16}, k_{l-15}, ..., k_{l-1}\}$ and $K = K_0^c||K_1^c||...||K_{Lenght-1}^c$. Then we calculate $K_L = K_0^c \oplus K_1^c \oplus ... \oplus K_{Lenght-1}^c$. If $K_L = 0$ then K_L is chosen as 0xC5C3, i.e. $K_L = 0$xC5C3. When generating a round keys K_i^c, $i = \overline{Lenght...16n+47}$, we used transformation *SubBytes16()* and *RotWord16()*, here *SubBytes16()*-is transformation 16-bit subblock into S-box and $SubBytes16(X) = S(sb_0(X))||S(sb_1(X)))$, *RotWord16()*-cyclic shift to the left of 1 bit of the 16-bit subblock. When the condition $imod3 = 1$ is true, then the round keys are computed as $K_i^c = SubBytes16(K_{i-Lenght+1}^c) \oplus SubBytes16(RotWord16(K_{i-Lenght}^c)) \oplus Rcon[imod16] \oplus K_L$, otherwise $K_i^c = SubBytes16(K_{i-Lenght}^c) \oplus SubBytes16(K_{i-Lenght+1}^c) \oplus K_L$. After each round key generation the value K_L is cyclic shift to the left by 1 bit.

Decryption round keys are computed on the basis of encryption round keys and decryption round keys of the output transformation associate with of encryption round keys as follows:

$$\begin{aligned}
&(K_{16n}^d, K_{16n+1}^d, K_{16n+2}^d, K_{16n+3}^d, K_{16n+4}^d, K_{16n+5}^d, K_{16n+6}^d, K_{16n+7}^d, K_{16n+8}^d, K_{16n+9}^d, K_{16n+10}^d,\\
&\quad K_{16n+11}^d, K_{16n+12}^d, K_{16n+13}^d, K_{16n+14}^d, K_{16n+15}^d)\\
=\ &(-K_0^c, -K_1^c, -K_2^c, -K_3^c, -K_4^c, -K_5^c, -K_6^c, -K_7^c, (K_8^c)^{-1}, (K_9^c)^{-1}, (K_{10}^c)^{-1},\\
&\quad (K_{11}^c)^{-1}, (K_{12}^c)^{-1}, (K_{13}^c)^{-1}, (K_{14}^c)^{-1}, (K_{15}^c)^{-1}).
\end{aligned}$$

For example, if the number of rounds n is 10 the formula is as follows:

$$\begin{aligned}
&(K_{160}^d, K_{161}^d, K_{162}^d, K_{163}^d, K_{164}^d, K_{165}^d, K_{166}^d, K_{167}^d, K_{168}^d, K_{169}^d, K_{170}^d, K_{171}^d, K_{172}^d, K_{173}^d, K_{174}^d, K_{175}^d)\\
=\ &(-K_0^c, -K_1^c, -K_2^c, -K_3^c, -K_4^c, -K_5^c, -K_6^c, -K_7^c, (K_8^c)^{-1}, (K_9^c)^{-1}, (K_{10}^c)^{-1},\\
&\quad (K_{11}^c)^{-1}, (K_{12}^c)^{-1}, (K_{13}^c)^{-1}, (K_{14}^c)^{-1}, (K_{15}^c)^{-1}).
\end{aligned}$$

Likewise, the decryption round keys of the first, second, third, and n–round associates with the encryption round keys as follows:

$$\begin{aligned}
&(K_{16(i-1)}^d, K_{16(i-1)+1}^d, K_{16(i-1)+2}^d, K_{16(i-1)+3}^d, K_{16(i-1)+4}^d, K_{16(i-1)+5}^d, K_{16(i-1)+6}^d, K_{16(i-1)+7}^d, K_{16(i-1)+8}^d,\\
&\quad K_{16(i-1)+9}^d, K_{16(i-1)+10}^d, K_{16(i-1)+11}^d, K_{16(i-1)+12}^d, K_{16(i-1)+13}^d, K_{16(i-1)+14}^d, K_{16(i-1)+15}^d)\\
=\ &(-K_{16(n-i+1)}^c, -K_{16(n-i+1)+1}^c, -K_{16(n-i+1)+2}^c, -K_{16(n-i+1)+3}^c, -K_{16(n-i+1)+4}^c, -K_{16(n-i+1)+5}^c,\\
&\quad -K_{16(n-i+1)+6}^c, -K_{16(n-i+1)+7}^c, (K_{16(n-i+1)+8}^c)^{-1}, (K_{16(n-i+1)+9}^c)^{-1}, (K_{16(n-i+1)+10}^c)^{-1},\\
&\quad (K_{16(n-i+1)+11}^c)^{-1}, (K_{16(n-i+1)+12}^c)^{-1}, (K_{16(n-i+1)+13}^c)^{-1}, (K_{16(n-i+1)+14}^c)^{-1}, (K_{16(n-i+1)+15}^c)^{-1}),\\
&\quad i = \overline{1...n}
\end{aligned}$$

5 Results

Using the transformations SubBytes(), ShiftRows(), MixColumns(), AddRoundKey() of the encryption algorithm AES as the round function networks PES16-1 and RFWKPES16-1 we developed encryption algorithm AES-PES16-1 and AES-RFWKPES16-1. In the algorithm, the number of rounds and key's length is variable and the user can select the number of rounds and the key's length in dependence of the degree of secrecy of information and speed encryption.

As in the encryption algorithms based on the Feistel network, the advantages of the encryption algorithms AES-PES16-1 and AES-RFWKPES16-1 are that, when encryption and decryption process used the same algorithm. In the encryption algorithms AES-PES16-1 and AES-RFWKPES16-1 in decryption process encryption round keys are used in reverse order, thus on the basis of operations necessary to compute the inverse. For example, if the round key is multiplied by the subblock, while decryption is is necessary to calculate the multiplicative inverse, if summarized, it is necessary to calculate the additive inverse.

It is known that the resistance of AES encryption algorithm is closely associated with resistance S-box, applied in the algorithm. In the S-box's encryption algorithm AES algebraic degree of nonlinearity deg = 7, nonlinearity $NL = 112$, resistance to linear cryptanalysis $\lambda = 32/256$, resistance to differential cryptanalysis $\delta = 4/256$, strict avalanche criterion SAC = 8, bit independence criterion BIC = 8.

In the encryption algorithm AES-PES16-1 and AES-RFWKPES16-1 resistance S-box is equal to resistance S-box's encryption algorithm AES, i.e., deg = 7, $NL = 112$, $\lambda = 32/256$, $\delta = 4/256$, SAC= BIC=8.

6 Conclusions

It is known that as a algorithms based of Feistel network, the resistance algorithm based on networks PES16-1 and RFWKPES16-1 closely associated with resistance round function. Therefore, selecting the transformations SubBytes(), ShiftRows(), MixColumns() and AddRoundKey() of the encryption algorithm AES, based on round function networks PES16-1 and RFWKPES16-1 we developed relatively resistant encryption algorithm.

References

[1] B. Bahrak and M. R. Aref, "A novel impossible differential cryptanalysis of AES," *Proceedings of the Western European Workshop on Research in Cryptology 2007*, pp. 152–156, 2007.

[2] B. Bahrak and M. R. Aref, "Impossible differential attack on seven-round AES-128," *IET Information Security*, vol. 2, no. 2, pp. 28–32, 2008.

[3] E. Biham, A. Biryukov, and A. Shamir, "Miss-in-the-middle attacks on IDEA, Khufu and Khafre," *6th Fast Software Encryption Workshop*, LNCS 1636, pp. 124–138, Springer, 1999.

[4] E. Biham and N. Keller, "Cryptanalysis of reduced variants of Rijndael," *Unpublished Manuscript*, 1999. (`http://madchat.fr/crypto/codebreakers/35-ebiham.pdf`)

[5] J. Borst, *Differential-linear Cryptanalysis of IDEA*, ESATCOSIC Technical Report 96/2, 1997. (`http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.49.5084&rep=rep1&type=pdf`)

[6] J. Borst, L. Knudsen, and V. Rijmen, "Two attacks on reduced IDEA (extended abstract)," in *Advances in Cryptology (Eurocrypt97)*, LNCS 1233, pp. 1–13, Springer, 1997.

[7] J. Chen, Y. Hu, and Y. Wei, "A new method for impossib le differential cryptanalysis of 8-round adanced encryption standard," *Wuhan Univeristy Journan of National Sciences*, vol. 11, no. 6, pp. 1559–1562, 2006.

[8] J. Chen, Y. Hu, and Y. Wei, "A new method for impossible differential cryptanalysis of 7-round advanced encryption standard," in *Proceedings of IEEE International Conference on Communications, Circuits and Systems*, vol. 3, pp. 1577–1579, 2006.

[9] J. Chen, Y. Hu, and Y. Zhang, "Impossible differential cryptanalysis of advanced encryption standard," *Science in China Series F: Information Sciences*, vol. 50, no. 3, pp. 342350, 2007.

[10] J. Cheon, M. Kim, K. Kim, J. Y. Lee, and S. Kang, "Improved impossible differential cryptanalysis of rijndael and crypton," in *proceedings of Information Security and Cryptology (ICISC'01)*, LNCS 2288, pp. 39–49, Springer, 2002.

[11] J. Daeman and V. Rijmen, "AES proposal: Rijndael, version 2," 1999. (`http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf`)

[12] J. Daemen, R. Govaerts, and J. Vandewalle, *Cryptanalysis of 2.5 Rounds of IDEA (Extended Abstract)*, ESAT-COSIC Technical Report 93/1, 1993.

[13] H. Demirci and A. Selcuk, "A meet-in-the-middle attack on 8-round AES," in *Proceedings of Fast Software Encryption*, LNCS 5806, pp. 116-126, Springer, 2008.

[14] N. Ferguson, J. Kelsey, S. Lucks, B. Schneier, M. Stay, D. Wagner, and D. Whiting, "Improved cryptanalysis of rijndael," in *Proceedings of Fast Software Encryption*, LNCS 1978, pp. 213-230, Springer, 2001.

[15] H. Gilbert and M. Minier, "A collision attack on 7 rounds of rijndael," in *Proceedings of the Third AES Candidate Conference (AES3)*, pp. 230-241, 2000.

[16] P. Hawkes, "Differential-linear weak key classes of IDEA," *Advances in Cryptology (Eurocrypt98)*, LNCS 1403, pp. 112-126, 1998.

[17] J. Kelsey, B. Schneier, and D. Wagner, "Key-schedule cryptanalysis of IDEA, g-DES, gost, safer and triple-DES," *Advances in Cryptology (Crypto96)*, LNCS 1109, pp. 237-251, Springer, 1996.

[18] J. Kim, S. Hong, and B. Preneel, "Related-key rectangle attacks on reduced AES-192 and AES-256," in *Proceedings of Fast Software Encryption*, LNCS 4593, pp. 225-241, Springer, 2007.
[19] L. R. Knudsen and V. Rijmen, *Truncated Differentials of IDEA*, ESATCOSIC Technical Report 97/1, 1997.
[20] X. Lai, "On the design and security of block ciphers," *Doctoral Theses, Hartung-Gorre*, 1992.
[21] X. Lai and J. L. Massey, "A proposal for a new block encryption standard," in *Advances in Cryptology (Eurocrypt90)*, LNCS 473, pp. 389-404, Springer, 1990.
[22] X. Lai, J. L. Massey, and S. Murphy, "Markov ciphers and differential cryptanalysis," in *Advances in Cryptology (Eurocrypt91)*, LNCS 547, pp. 17-38, Springer, 1991.
[23] J. Lu, O. Dunkelman, N. Keller, and J. Kim, "New impossible differential attacks on AES," in *Advances in Cryptology (INDOCRYPT'08)*, LNCS 5365, pp. 279–293, Springer, 2008.
[24] S. Lucks, "Attacking seven rounds of rijndael under 192-bit and 256-bit keys," in *proceedings of the Third AES Candidate Conference (AES3)*, pp. 215-229, 2000.
[25] W. Meier, "On the security of the IDEA block cipher," in *Advances in Cryptology (Eurocrypt93)*, LNCS 765, pp. 371-385, Springer, 1994.
[26] J. Nakahara, P. Barreto, B. Preneel, J. Vandewalle, and Y. Kim, "Square attacks on reduced-round pes and IDEA block ciphers," in *23rd Symposium on Information Theory*, pp. 187-195, 2002.
[27] National Institute of Standards and Technology, *Announcing the Advanced Encryption Standard (AES)*, Federal Information Processing Standards Publication 197, 2001. (`http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`)
[28] U. S. Department of Commerce/National Institute of Standards and Technology, *Data Encryption Standard (DES)*, Federal Information Processing Standards Publication 46-3, 1079. (`http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf`)
[29] R. Ch-W. Phan, "Impossible differential cryptanalysis of 7-round advanced encryption standard (AES)," *Information Processing Letters*, vol. 91, no. 1, pp. 33-38, 2004.
[30] G. N. Tuychiev, "About networks RFWKPES16-8, RFWKPES16-4, RFWKPES16-2 and RFWKPES16-1, created on the basis network PES16-8," *Ukrainian Information Security Research Journal* (in print).
[31] G. N. Tuychiev, "About networks IDEA16-4, IDEA16-2, IDEA16-1, created on the basis of network IDEA16-8," *Compilation of Theses and Reports Republican Seminar on Information Security in the Sphere Communication and Information. Problems and Their Solutions*, 2014.
[32] G. N. Tuychiev, "About networks IDEA8-2, IDEA8-1 and RFWKIDEA8-4, RFWKIDEA8-2, RFWKIDEA8-1 developed on the basis of network IDEA8-4," *Uzbek Mathematical Journal*, no. 3, pp. 104–118, 2014.
[33] G. N. Tuychiev, "About networks pes8-2 and pes8-1, developed on the basis of network pes8-4," in *Transactions of the International Scientific Conference on Modern Problems of Applied Mathematics and Information Technologies-Al-Khorezmiy*, vol. 2, pp. 28–32, 2014.
[34] G. N. Tuychiev, "About networks RFWKPES8-4, RFWKPES8-2, RFWKPES8-1, developed on the basis of network PES8-4," in *Transactions of the International Scientific Conference on Modern Problems of Applied Mathematics and Information Technologies-Al-Khorezmiy*, vol. 2, pp. 32–36, 2014.
[35] G. N. Tuychiev, "New encryption algorithm based on network RFWKPES8-1 using of the transformations of the encryption algorithm AES," *International Journal of Multidisciplinary in Cryptology and Information Security*, vol. 3, no. 6, pp. 31–34, 2014.
[36] G. N. Tuychiev, "About networks pes16-4, pes16-2 and pes16-1, created on the basis network pes16-8," *Ukrainian Information Security Research Journal*, vol. 17, no. 1, pp. 53–60, 2015.
[37] G. N. Tuychiev, "New encryption algorithm based on network IDEA16-1 using of the transformation of the encryption algorithm AES," *International Journal of Information Technology*, vol. 3, pp. 6–12, 2015.
[38] G. N. Tuychiev, "New encryption algorithm based on network IDEA8-1 using of the transformation of the encryption algorithm AES," *International Journal of Computer Science*, vol. 3, pp. 1–6, 2015.
[39] G. N. Tuychiev, "New encryption algorithm based on network pes8-1 using of the transformations of the encryption algorithm AES," *International Journal of Multidisciplinary in Cryptology and Information Security*, vol. 4, no. 1, pp. 1–5, 2015.
[40] G. N. Tuychiev, "New encryption algorithm based on network rfwkIDEA8-1 using transformation of AES encryption algorithm," *International Journal of Computer Networks and Communications Security*, vol. 2, no. 3, pp. 43–47, 2015.
[41] W. Zhang, W. Wu, and D. Feng, "New results on impossible differential cryptanalysis of reduced AES," in *proceedings of ICISC 2007*, LNCS 4817, pp. 239–250, Springer, 2007.
[42] W. Zhang, W. Wu, L. Zhang, and D. Feng, "Improved related-key impossible differential attacks on reduced-round AES-192," in *Proceedings of Selected Areas in Cryptography*, LNCS 4356, pp. 15–27, Springer, 2007.

Tuychiev Gulom is a candidate technical Sciences (Ph.D.), National University of Uzbekistan. He received Pd.D. degree in specialty mathematic from the National University of Uzbekistan. He had published more than 50 scientific articles. He current research interests include block ciphers, Boolean functions.

Fuzzy Based Energy Aware Routing Protocol with Trustworthiness for MANET

Deverajan Ganesh Gopal[1] and R. Saravanan[2]

(Corresponding author: Deverajan Ganesh Gopal)

School of Computing Science and Engineering, Vellore Institute of Technology, India[1]
School of Information Technology and Engineering, Vellore Institute of Technology, India[2]
(Email: ganeshgopal@vit.ac.in and rsaravanan@vit.ac.in)

(Received Apr. 20, 2014; revised and accepted July 5 & Dec. 12, 2014)

Abstract

Of late, trust based routing is the most researchable topic in Mobile Ad hoc Networks (MANET). Trust level management of nodes is a critical issue, because of the infrastructure less nature of MANET. Each node is involved in the process of packet transmission and hence each node must be trustworthy. In this work, we propose a new protocol namely Adaptive Fuzzy DoT Threshold Routing Algorithm (AFTRA), which takes into account the Degree of Trust (DoT), connectivity and the energy levels. AFTRA provides all possible routes from the source to the destination. The best route is selected by considering three aspects hop count, trust values and energy. Among these, connectivity (hop count) is given more importance and based on this the threshold value is calculated for each route. Energy is given lesser priority as we concentrate on trust. The final route obtained can be declared as the shortest path available with the most trustworthy nodes. Thus, the most trustworthy and shortest route is obtained by using the proposed approach. We compare our approach with the existing protocols in terms of Packet Delivery Ratio, End-to-End Delay, Routing overhead etc. The overall performance of this system can be quoted as 98%. The proposed method outperforms many other existing systems.

Keywords: AFTRA; DoT; Hop Count; MANET; Trustworthy.

1 Introduction

Of late, trust based routing is the most researchable topic in Mobile Ad hoc Networks (MANET). Trust level management of nodes is a critical issue, because of the infrastructure less nature of MANET. Every node behaves as router and transmits messages through multiple hops, in order to reach the destination. Thus, every node is involved in the process of packet transmission and hence every node must be trustworthy.

During the transmission of a packet from the source to the destination, each node in the chosen route needs to be trustworthy. MANET is distributed widely and hence the network topology of MANET is not constant [23].

Many researches focus on trust based model now-a-days. The challenging task is to provide a trust based model and choosing the factors for it [28]. Due to this, MANET is prone to several security threats. Security is a risk factor, as any entity in the middle can infer the information. So, it is necessary to concentrate on selecting an effective route and the intrusion should be detected.

External attacks are carried out by the external nodes by providing wrong route information. On the other hand, internal attacks are done by the nodes within the network. The attacks are caused by selfish and malicious nodes. Selfish nodes conserve their energy and do not cooperate with the networks. Selfish nodes do not damage other nodes, whereas malicious nodes affect other nodes. Detecting internal attacks is more complex than detecting external attacks.

In MANET, every node is involved in packet transmission and the selfish or malicious nodes behave oddly, which affects the entire system. There is an inevitable need to categorize between normal and abnormal (selfish or malicious) nodes. In order to detect selfish nodes, trust factor is employed.

In this work, we propose a system that is based on the Degree of Trust (DoT) of every node and the malicious nodes are detected too. The trust value is calculated for every node and the nodes are categorized as trustworthy, partially trustworthy and malicious nodes. The main aim of this work is to improve security, packet delivery ratio, throughput etc.

This work considers the degree of selfishness, connectivity and energy in order to calculate DoT. The degree of selfishness is determined by the forwarding history evidence. Forwarding history evidence maintains the message forwarding history.

In and out-ratio of the forwarding history evidences is calculated. In normal case, the in-ratio must be equal to the out-ratio. If the in-ratio is twice the out ratio, then the degree of selfishness will be 0.5. If the out- ratio is zero, then the node is considered to be completely selfish or it could be because of energy drop out. Connectivity is the term that indicates the number of intermediate nodes in between the source and the destination. Energy is the battery backup of a node and the energy diminishes when the node transmits many messages.

This work considers the degree of selfishness of nodes as a selfish node will not forward any messages for the sake of other nodes, in order to conserve its energy, which in turn reduces the trust.

Connectivity is also considered in routing because if the destination node is far away from the source node with several non-trustworthy nodes, then the level of trust reduces. The reason why energy is considered in routing is because, if the energy is low, then there may be.

The major contribution of this work is towards handling network attacks, as most of the existing routing protocols are meant for coping up with the dynamic network topology and lag behind in handling network attacks. In this work, we propose a system based on DoT that effectively chooses a route based on the Degree of Trust (DoT) of each node and also detects the malicious nodes. This system improves security, packet delivery ratio, throughput and also saves energy. We compare our system with the existing protocols in terms of Packet delivery ratio, end-to-end delay, routing overhead, throughput, classification accuracy, detection time and detection rate.

The remaining work is organized as follows. Section 2 presents the review of literature, Section 3 deals with DoT calculation, route discovery and selection. Section 4 presents the principle and methodology of our system. Experimental analysis is presented in Section 5 and finally, concluding remarks is provided.

2 Background

The existing routing protocols focus on the mobile network topology and lag behind in handling network attacks. In this section, the review of literature is presented for trust based and neuro fuzzy routing.

2.1 Trust based Routing Protocols

In MANET to make a safe packet forwarding here introduced concept of trust and QOS metric estimation into launching a trust-based QOS model, they estimate the trust degree among nodes from direct trust calculation of direct observations and indirect trust computation by neighbor's references. In NS2 they implement this algorithm based on Ad hoc On-demand Distance Vector and compare its performance with Ad hoc On-demand Distance Vector, Watchdog-DSR and QAODV the proposed scheme can prevent attacks from malicious nodes and improve the security performance of the whole network [23].

Evaluating the trust worthiness of other node without central authorities which will improves the security of the ad hoc networks, here a theoretic framework is presented, to measure trust and build novel model with several decision factors. It is incorporated to reflect trust relationship's complexity uncertainty in different angles. Setting up fuzzy analytic hierarchy based on the weight of factors, which makes the model a better rationality they proposed a novel reactive trust -based multicast routing protocol is proposed [28].

MANET has deployed geographically limited without well-established infrastructure, because of its openness topology the MANET's is open to attackers for finding vulnerability and malicious attacks from the node. To reduce hazard and enhance the security in the network they presented a mobile trust prediction model to evaluate the trustworthiness of nodes, which it is based on the historical behavior of the node and future behavior by fuzzy logic rules prediction. Integrated with trust prediction model into the source routing mechanism, and termed as TSR (Trust based Source Routing Protocol) provides flexible approach to choose shortest route and security requirements of data packets transmission [27]. Security is the major important role in executing mobile ad hoc networks (MANET) for communication in an adverse environment, finding the misbehavior user in the network and report's it anonymously. Here TEAP (Trust-enhanced anonymous on-demand routing protocol) I proposed to restrain misuse of anonymity it in two methods, TEAP protocol is designed based on the trapdoor to find the misbehavior user anonymously in the Network [8]. Here they Presents a trust prediction model to evaluate the trustworthiness of nodes, which is based on the nodes' historical behaviors and fuzzy logic rules prediction method. As an application of the proposed trust model, extended from the ad-hoc on-demand multi-path distance vector routing (AOMDV) protocol, a trust-based reactive multi-path routing protocol, named as ad hoc on-demand trusted multi-path distance vector routing (AOTMDV) protocol, is proposed for MANETs [25].

In [20], every device is protected with Trusted Platform Module (TPM), which is a cryptographic protocol. This hardware chip makes it possible to fix its local state and also can predict the states of remote systems too. Thus, malicious devices are automatically detected and are not allowed to take part in the network. The work proposed in [22] presents a reputation system along with the trust system. In this work, both personal and general evidences are used to decide packet transmission. A trust based threshold cryptographic revocation scheme is proposed in [5], in which the master private key is divided into n number of pieces with respect to a random polynomial and this scheme is claimed to be robust. . In [17], the authors present a light-weight trust-based routing protocol. It is light-weight in the sense that the intrusion detection system (IDS) used for estimating the trust that one node has for another, consumes limited computational resource.

The work proposed in [2] analyzes the trust level of all nodes by employing entropy based trust system. Many routing protocols exist, but still most of the protocols are meant for coping up with the mobile network topology and lag behind in handling network attacks. The attacks can be external or internal [7] and [26]. A survey on trust management for mobile ad hoc networks is presented in [4, 13].

A scheme for trust certificate distribution is provided and is based on swarm intelligence. An attack resilient Hybrid Trust Management Framework (HTMF) is presented in [15].MANET security issues and trusted computing is presented in [6]. MANET's are increasingly reaching many other applications in areas such as intelligent transportation systems and fault-tolerant mobile sensor grids [3]. Secure ad hoc net-works have to meet five security requirements: confidentiality, integrity, authentication, non-repudiation and availability [1].

A Fuzzy based Ad hoc On-demand Distance Vector (FAODV) routing protocol is proposed in [18] a threshold value is set for trust verification and the trust value is evaluated by imposing fuzzy rules. Trust and reputation systems represent a significant trend in decision support for Internet mediated service provision [12] and [14].

An enhancement of DSR is presented here and is named Trusted DSR and proposed in [10]. The performance of three different protocols namely Trusted AODV, DSR and TORA were compared and the results concluded that Trusted AODV works well with a stable throughput [21] and [11]. Some of the routing protocols paired with cryptography are proposed in [29] and [9], Trust based intrusion detection system is presented in [19], in which three types of trust are defined. Un-slotted carrier-sense multiple access protocol is employed and it avoids collision [24] for packet transmission.

3 Trust Based Model

The term 'trust' can be defined as the value that indicates the real behavior of a node. In [23], trust is defined as the relationship between two neighboring entities. Using this trust value, an effective route can be predicted.

This can be attained by choosing the path with the least non-trustworthy nodes. In this work, we use three different parameters for DoT calculation.

They are degree of selfishness, connectivity and energy. These three parameters can effectively evaluate the trust level. The optimal route is selected by this trust value. The trust model is shown in Figure 1.

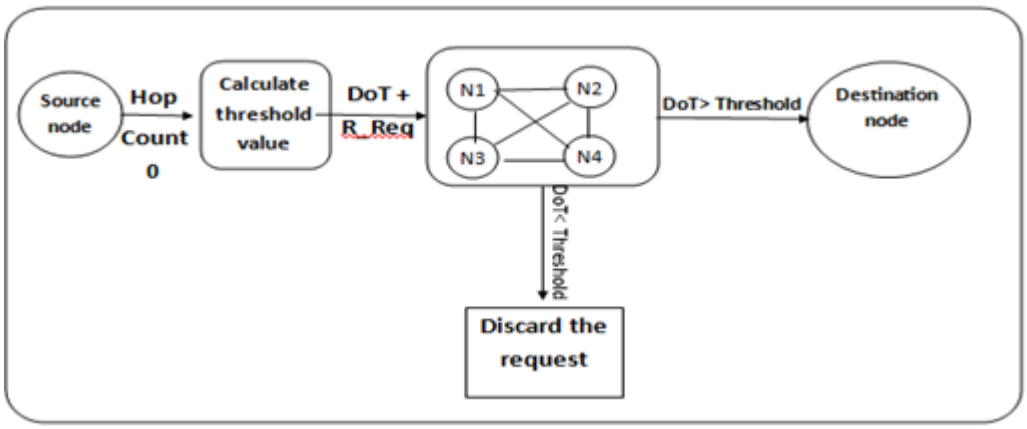

Figure 1: Trust model

3.1 Overview of Proposed Scheme

In this work, we consider the DoT of both the nodes and the complete path as well. If many paths are available for packet transmission from the source to the destination, then the route with the maximum number of DoT and the least number of hops is chosen.

A threshold value is fixed for all the three parameters, in order to classify the nodes as trustworthy, partially trustworthy and malicious. Then, all the three values are put together and the most effective route is selected.

The degree of trust of each node is computed and stored in a buffer for each time interval and are over-written in order to save memory. The nodes are arranged in ascending order, such that the normal nodes are placed first and stored in the table.

We propose to consider only the first half of the table, so as to avoid the nodes that are completely selfish. When a source node attempts to forward a packet to the destination, then the routes are computed by considering the connectivity and energy. Connectivity is the number of intermediate nodes present between the source and the destination. The lesser the value of connectivity, the shorter is the route. Energy of a node is also taken into account, so as to avoid the risk of node failure or node detachment because of diminishing energy.

The connectivity (hop count) of all the possible routes is arranged in ascending order, so that the shortest routes are primarily placed. Then, the frequency of nodes with the least degree of selfishness is calculated for every route. It is arranged in descending order in order to select the route with the most normal nodes. This is followed by energy calculation of the nodes in the first half of the arranged routes.

The energy value for all the nodes in each route are added, and if the sum of energies is equal to the number of nodes involved in the route, then it are the first best route. If the sum of energies is equal to three-fourth of the number of nodes involved in the route, then it is the second best route. If the sum of energies is equal to half of the number of nodes involved in the route, then it is the third best route. When the sum of energies is equal to one-fourth of the number of nodes in the route, then the route is declared as the worst route. The overall flow AFTRA is given in Figure 2.

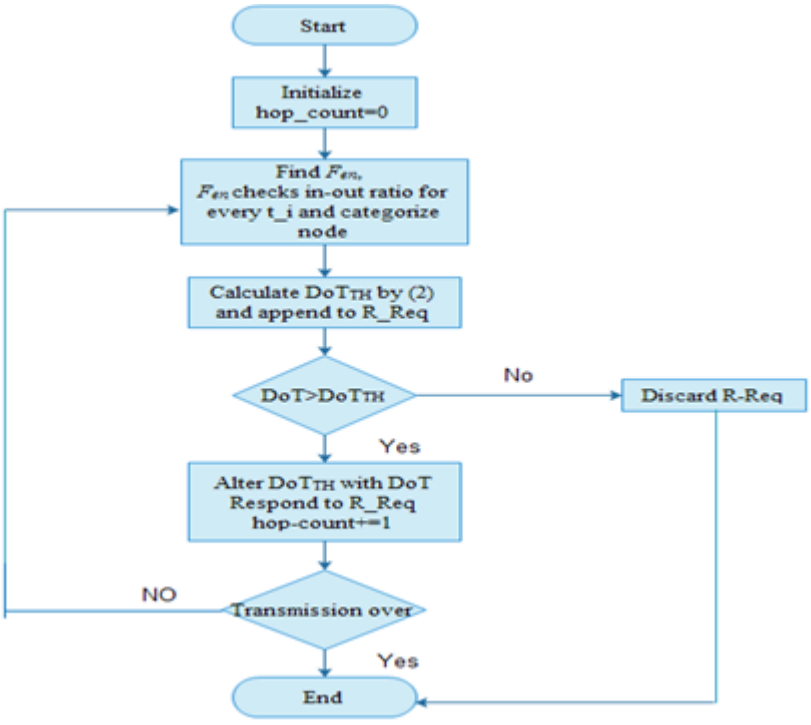

Figure 2: Overall flow of AFTRA

3.2 DoT Evaluation

In this work, DoT is calculated by taking the in-out ratio of nodes. The degree of trust is determined by the forwarding history evidence. Forwarding history evidence maintains the message forwarding history. That is, when node1 sends a message through node 3, the evidence would contain the sender id, receiver id, message, signature of

the source and timestamp [27]. These details are stored in local memory of a node and are cleansed over time. The data in this table determines the behavior of every node.

The in and out-ratio of the forwarding history evidences is calculated. In normal case, in-ratio must be equal to the out-ratio. If the in-ratio is twice the out ratio, then the degree of selfishness will be 0.5. If the out- ratio is zero, then the node is considered to be completely selfish or it could be because of energy drop out.

Let γ be the in-ratio of packets and μ be the out-ratio of the packets. If $\gamma = \mu$, then the node is trustworthy, that is, it forwards all the received packets and actively participates in the network. This type of node renders its fullest cooperation to the network.

If $\gamma = \mu/2$, then the node is partially trustworthy. This type of node will forward the packets sometimes and not always. The behavior of such nodes varies with respect to time. If the value of $\mu = 0$, then the node is malicious and it may affect the entire network.

Table 1: Sample DoT table

Degree	DoT	Description
1	$\gamma = \mu$	Trustworthy Node
2	$\gamma = \mu/2$	Partially Trustworthy Node
3	$\mu = 0$	Malicious Node

Table 2: Energy value table

Case	Energy value	Description
1	1	Full energized node
2	0.75	Pretty good energy
3	0.5	Half energized node
4	0.25	Poor energized node
5	0	Energy drained node

Table 3: Sample routing table

Destination	Destination Sequence	Adjacent Hop	Hop Count	Trust Value	Energy Value
A	A	A	1	0.5	0.7
E	B	A, B	3, 4	0.8	0.6

Every one-hop neighbor has the provision to access the energy value table. If any node finds that energy value is 1, then it broadcasts the message submit FHE to all the nodes present in the network. This node collects the Forward History Evidence of every node and declares the node as trustworthy node, partially trustworthy node or malicious node and is stored in buffer for future reference. The computed DoT is provided as the input for the routing algorithm.

3.3 AFTRA: Adaptive Fuzzy DoT Threshold Routing Algorithm

In the first phase, the input values are transformed to membership functions and are presented below.

Fuzzy DoT Threshold

In this work, we classify the DoT value into three groups namely minimum (min), medium (mid), maximum (max). Let $DoT_i = \{0, 0.5, 1\}$ and is evident that $Min\_DoT_i = \{0\}$, $Mid\_DoT_i = \{0.5\}$ and $Max\_DoT_i = \{1\}$. The membership value for i^{th} node is given by

$$\mu_i(DoT_i) = max(Min\_DoT_i, Mid\_DoT_i, Max\_DoT_i). \quad (1)$$

The threshold value DoT_{TH} and the membership value is medium among the neighboring nodes and is given by

$$\mu_{TH}(DoT_{TH}) =^{mid}_{(0 \le i \le 1)} \{\mu_i(DoT_i)\}. \quad (2)$$

The value obtained by this is the threshold value and is passed to other nodes, when sending route request R_Req. The algorithm for routing based on DoT is presented below.

Step 1. Let 'S' and 'D' be the Source and the Destination nodes, initialize the hop count with value 0.

Step 2. Calculate the threshold value based on Equation (2). Append the DoT_{TH} along with R_Req and flood the route request to all the neighboring nodes, in order to find the route from the source node to the destination node.

Step 3. On receiving the route request R_Req, every node checks the DoT_{TH} and compares the value with its own DoT. If its DoT is greater than the threshold value, then it participates in the packet transmission, else it discards the route request.

Step 4. The intermediate node replies to the route request by changing the DoT_{TH} with its own DoT value, if its connection is made possible to the destination node.

Step 5. The hop count is incremented by 1.

Step 6. Steps 3 through 5 is repeated until the packet is sent from the source node to the destination node.

The outcome of this algorithm is the formulation of effective route that weeds out all the non-trustworthy nodes along the path. This algorithm yields a trustworthy route and thus the packet is sent all at once. Using the above algorithm the total number of possible routes is predicted and the hop count of every route is stored in buffer. The optimal route is selected by following the steps presented below.

1) Arrange hop_count of all routes in ascending order, in order to obtain the shortest route.

2) Calculate the occurrence frequency of trustworthy nodes (trustworthy nodes can be obtained from buffer that has arranged nodes in ascending order with respect to DoT), that is the nodes with DoT greater or equal to DoT_{TH} and arrange it in descending order. The threshold value is fixed by

$$OFT_{Th} = \frac{Total\_number\_of\_nodes\_in\_the\_route}{2} \tag{3}$$

3) The energy value for all the nodes in each route are added and if the sum of energies is equal to the number of nodes involved in the route, then it are the first best route.

   a. If the sum of energies is equal to three-fourth of the number of nodes involved in the route, then it is the second best route.

   b. If the sum of energies is equal to half of the number of nodes involved in the route, then it is the third best route.

   c. When the sum of energies is one-fourth of the number of nodes in the route, then the route is declared as the worst route.

We give importance to hop count and based on this, OF_{Th} is calculated for every route. Energy is given lesser priority as we concentrate on trust. The final route from the source to the destination can be declared as the shortest path available with the most trustworthy nodes. Therefore, the most trustworthy and the shortest route is obtained and thus the target is achieved. The malicious nodes are weeded out by our algorithm.

The overall flow of the work is presented below.

All the possible routes can be figured out with the algorithm presented above and the optimal route is selected by the steps given below.

In this work, we concentrate more on finding the shortest route with maximum trustworthy nodes and energy is given less priority. To exemplify this concept consider Figure 2.

In Figure 3, node A is the source and D is the destination. The possible routes obtained by the algorithm are $A \to B \to C \to D$, $A \to E \to D$, $A \to F \to G \to D$. From all these routes, the route with the shortest path and most number of trustworthy nodes is the optimal route and is justified by the following calculation. On calculating the hop_count, route 2 is the shortest. When arranging the routes in ascending order with respect to hop_count, the result is $A \to E \to D$, $A \to B \to C \to D$ and $A \to F \to G \to D$. This step is followed by the calculation of occurrence frequency of trustworthy nodes, which can be obtained from the buffer. OFT of $A \to E \to D$ is 1, $A \to B \to C \to D$ is 2 and $A \to F \to G \to D$ is 3 respectively. Thus, even though $A \to E \to D$ is the shortest route the optimal route is $A \to F \to G \to D$.

S_{ev} is the sum of energy values of all nodes in a route. S_{ev} of $A \to E \to D$ is 1, $A \to B \to C \to D$ is 1.4 and $A \to F \to G \to D$ is 1.6 respectively. In this case, the first best route is not present as S_{ev} does not equal the node count in any route. In any case, $A \to F \to G \to D$ is the optimal route, with maximum S_{ev}.

Algorithm 1: Overall Flow of AFTRA

Initialize $hop\_count = 0$;
Find the full_energized node F_{en};
F_{en} checks in-out-ratio;
If $\gamma = \mu$ then node is trustworthy;
If $\gamma = \mu/2$ then node is partially trustworthy;
Else malicious node;
For every t_i (time_interval)
Calculate 3 and store in buffer;
Calculate DoT_{TH} by Equation (2) and append to R_Req;
Node compares its DoT with DoT_{TH};
If $DoT > DoT_{TH}$ then
Alter DoT_{TH} with DoT;
Respond to R_Req;
Else discard R_Req;
$hop\_count+ = 1$;
Repeat until the packet is transmitted;

Algorithm 2: Optimal Route

Calculate hop_count of all possible routes HCR;
Arrange HCR in ascending order;
Obtain type of node from buffer;
Calculate OFT of trustworthy nodes in every route;
Arrange OFT in descending order;
Add energy value S_{ev} of all nodes in every route;
If S_{ev} = number of nodes in route node_count then
Declare it as first_best_route;
Else if $S_{ev} = 3/4\times$ node_count then
Declare it as second_best_route;
Else if $S_{ev} = 1/2\times$ node_count then
Declare it as third_best_route;
Else declare it as worst_route;
End;

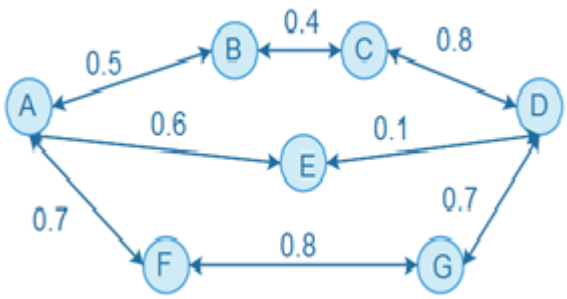

Figure 3: DoT of nodes

4 Experimental Analysis

The proposed work is compared with Trust based Source Routing (TSR) which has already been compared with DSR and TDSR [23]. So, in this work we compare TSR with our proposed algorithm in terms of Packet delivery ratio, end-to-end delay, routing overhead, throughput, classification accuracy, detection time and detection rate. Our algorithm outperforms the TSR and is shown in graphs. We have employed NS-2 for simulation. Un-slotted carrier-sense multiple access protocol is employed and it avoids collision [7] for packet transmission. We distribute

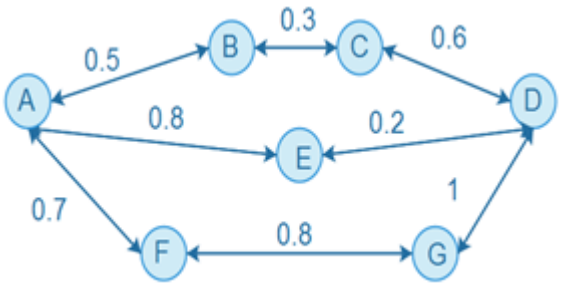

Figure 4: Energy values of nodes

30 nodes in a 1000 by 1000 meter. The transmission radius of all nodes for a hop is given as 250 meters.

Table 4: Simulation parameters

Simulation Parameters	Value
Simulation Time	250 sec
Dimension	$1000 \times 1000m$
Node Count	30
Traffic Nature	Constant Bit Rate
Transmission Radius	$250m$
Packet Size	512 bytes

4.1 Performance Analysis

Performance metrics such as Packet delivery ratio, end-to-end delay, routing overhead, throughput, classification accuracy, detection time and detection rate are used to evaluate the performance of our algorithm and the graphs of these performance metrics are presented from Figure 5 - Figure 11. FTDSR1 and FTDSR2 are proposed in the same work but with different characteristic feature. FTDSR1 pays more attention towards control packets and FTDSR2 focuses on data packets. On result analysis, it is proved that control packets are more important than data packets in MANET.

1) Packet Delivery Ratio (PDR): PDR is the ratio of packets that are successfully sent to the destination node from the source node and are computed by Equation (4) and the results are presented in Figure 1.

$$PDR = \frac{\sum Number\_of\_Packets\_Received}{\sum Number\_of\_packets\_sent} \times 100. \tag{4}$$

Our system shows 90% packet delivery ratio, which is far better than FTDSR1. Packet delivery ratio drops to 83% when the node count increases to 300. Our system shows a good range of PDR because of the selection of optimal path.

2) End-to-End Delay (EE_{dl}): The average time taken by the packets to reach the destination node from the source node and is calculated by

$$EE_{dl} = \frac{\sum(Arrival\_Time - Sent\_Time}{)} \sum Number\_of\_Connections \tag{5}$$

Our proposed scheme proves least end-to-end delay and outperforms all the protocols. The end-to-end delay of our system starts from 0.01seconds for 50 nodes and 2.2 seconds for 300 nodes. As the proposed AFTRA focuses more on connectivity, end-to-end delay is considerably minimized.

3) Routing Overhead: The ratio of control packet to the data packet count and is given by

$$Rt_o = \frac{Number\_of\_control\_packets}{Number\_of\_data\_packets} \tag{6}$$

Routing overhead of our proposed scheme is lesser than all the protocols being compared. Routing overhead is minimized because of the simple trust predicting ability of the system.

4) Throughput: The amount of data transmitted per unit time and it is calculated by

$$Throughput = \frac{Size\_of\_the\_Packets(bits)}{Time\_Taken(sec)} \tag{7}$$

Throughput is directly proportional to the packet delivery ratio and thus the throughput of the proposed work is higher than the existing works. The proposed work can transmit 0.54 packets per second.

5) Classification Accuracy: The capability of differentiating the nodes as trustworthy, partially trustworthy and malicious. Classification accuracy of this work is about 98.5% and this is comparatively greater than the existing works.

6) Detection Time: The time taken to classify a node correctly and it is computed by

$$Detection\_Time = Ending\_Time - Starting\_Time \tag{8}$$

Our proposed work consumes lesser time to detect the node correctly and it ranges from 2.12 to 2.67 seconds.

7) Detection Rate: The percentage of packets routed correctly with the total number of routes available and it is calculated by

$$Detection\_Rate = \frac{Number\_of\_Correctly\_detected\_packets}{/} Total\_number\_of\_nodes \times 100. \tag{9}$$

The detection rate of this system ranges from 97.12 to 97.86% for 50 to 300 nodes.

Corresponding graphs for all the above performance metrics are presented below.

From the above graphs, it is evident that the proposed algorithm works well, when compared to the existing algorithm, in terms of Packet delivery ratio, end-to-end delay, routing overhead, throughput, classification accuracy, detection time and detection rate.

5 Conclusion

Most of the existing routing protocols are meant for coping up with the mobile network topology and lag behind in handling network attacks. In this work, we propose a framework that is based on DoT routing scheme that effectively chooses a route based on Degree of Trust (DoT) of every node and the malicious nodes are also detected. We give importance to hop count and based on this, threshold is calculated for every route. Energy is given lesser priority as we concentrate on trust. The final route from the source to the destination can be declared as the shortest path available with the most trustworthy nodes. Thus, the most trustworthy and shortest route is obtained. We compare our system with the existing protocols in terms of Packet delivery ratio, end-to-end delay, routing overhead, throughput, classification accuracy, detection time and detection rate. However, the complexity of maintaining many parameters can be handled in future, in association with reducing the complexity involved in route establishment.

References

[1] L. Abusalah, A. Khokhar, and M. Guizani, "A survey of secure mobile ad hoc routing protocols," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 4, pp. 78–93, 2008.

[2] M. Alattar, F. Sailhan, J. Bourgeois, "Trust-enabled link spoofing detection in MANET", in *32nd International Conference on Distributed Computing Systems Workshops*, pp. 237–244, 2012.

[3] M. Carvalho, "Security in mobile ad hoc networks," IEEE Security & Privacy, vol. 6, no. 2, pp. 72–75, 2008.

[4] J. H. Cho, A. Swami, and I. R. Chen, "A survey on trust management for mobile ad hoc network," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 4, pp. 562–583, 2011.

[5] H. Dahshan, F. Elsayed, A. Rohiem, A. Elmoghazy, J. Irvine, "A trust based threshold revocation scheme for MANETs", in *78th IEEE Vehicular Technology Conference*, pp. 1–5, 2013.

[6] N. Garg and R. Mahapatra, "MANET security issues," *International Journal of Computer Science and Network Security*, vol. 9, no. 8, pp. 241–246, Aug. 2009.

[7] K. Govindan and P. Mohapatra, "Trust computations and trust dynamics in mobile adhoc networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 2, pp. 279–298, 2012.

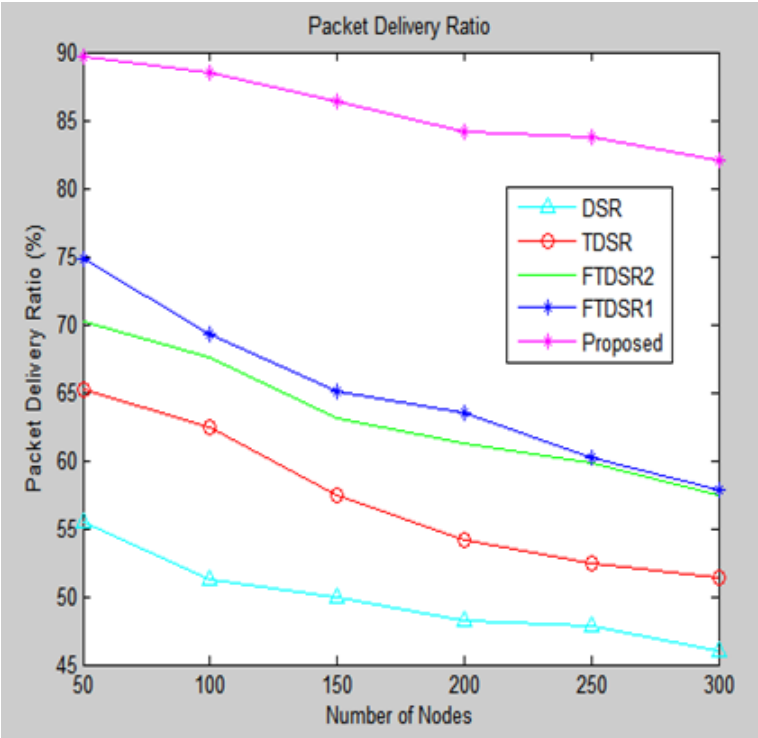

Figure 5: Packet delivery ratio analysis

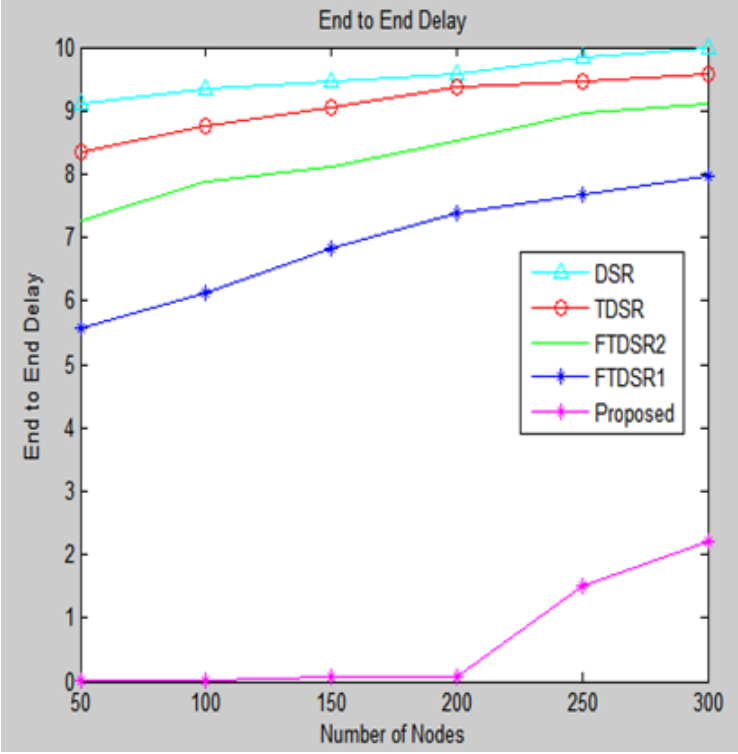

Figure 6: End-to-end delay analysis

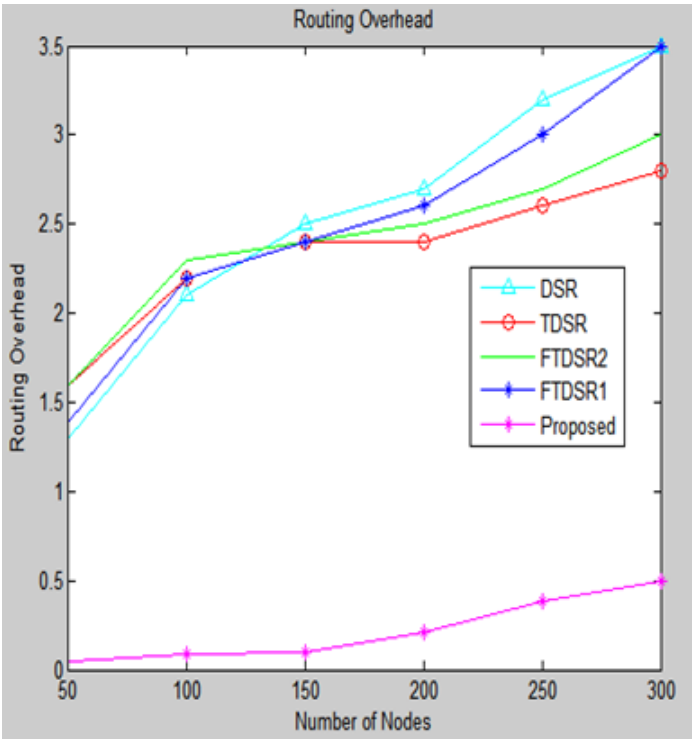

Figure 7: Routing overhead analysis

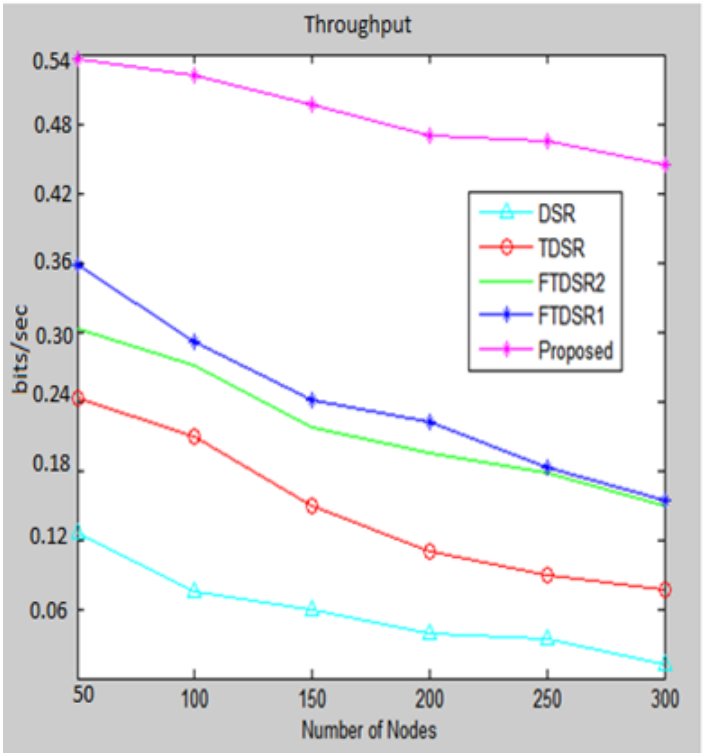

Figure 8: Throughput analysis

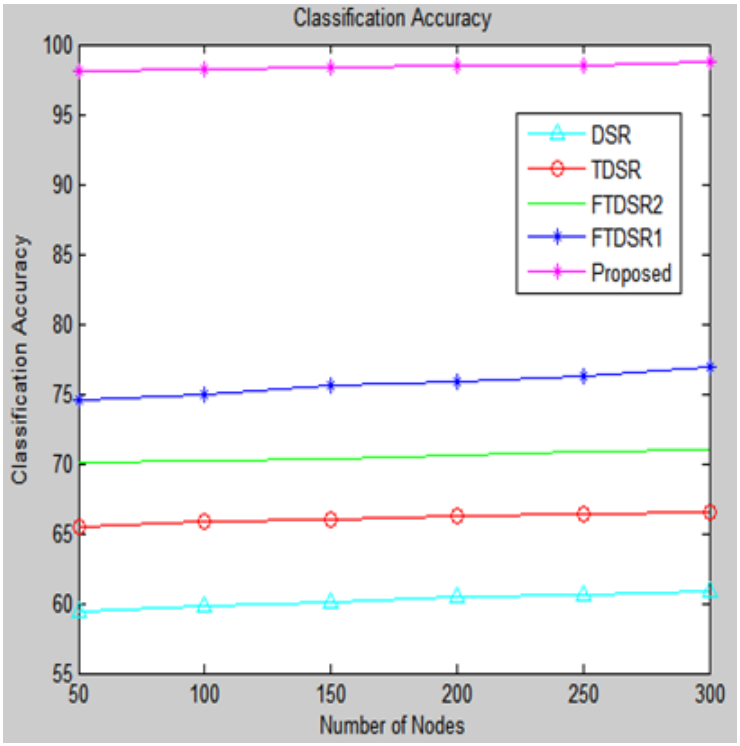

Figure 9: Classification accuracy analysis

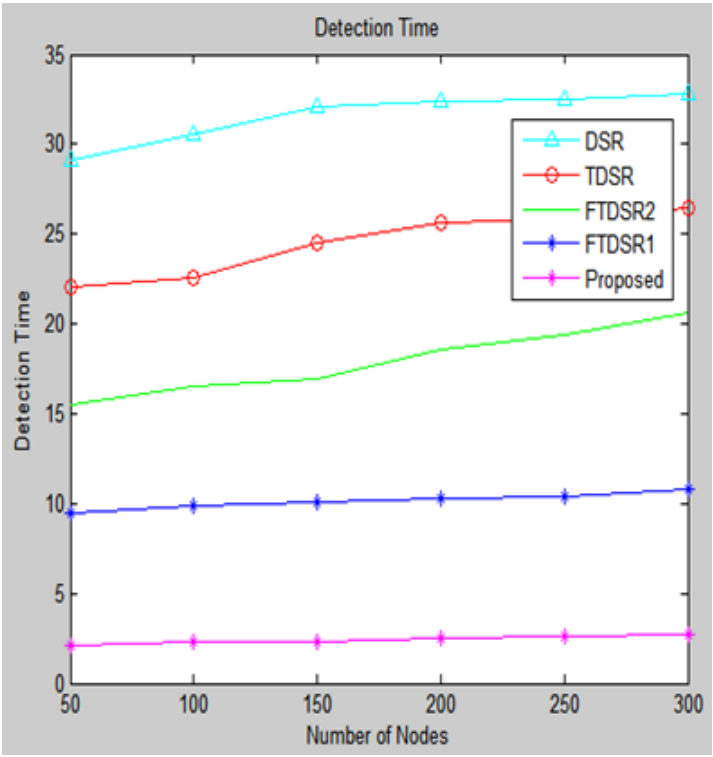

Figure 10: Analysis of detection time

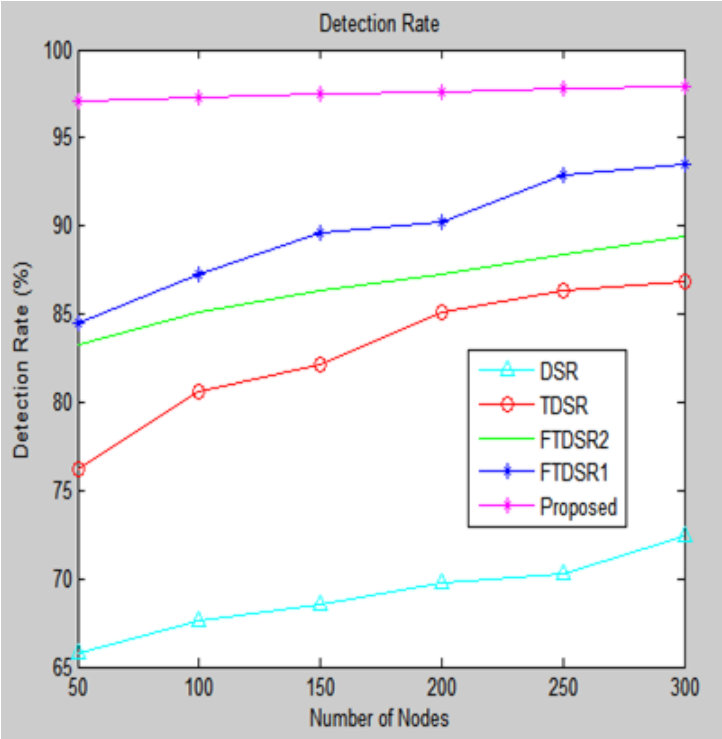

Figure 11: Detection rate analysis

[8] M. Gunasekeran, K. Premalatha, "TEAP: A trust-enhanced anonymous on-demand routing protocol for mobile ad hoc networks", *IET Information Security*, vol. 7, no. 3, pp. 203–211, Sept. 2013.

[9] Y. C. Hu, A. Perrig, D. B. Johnson, "Ariadne: a secure on-demand routing protocol for ad hoc networks", in *Proceedings of International Conference on Mobile Computing and Networking (Mobicom'02)*, pp. 12–23, 2002.

[10] C. D. Jensen, P. O. Connell, "Trust-based route selection in dynamic source routing," in *Proceedings of International Conference on Trust Management*, pp. 150–163, 2006.

[11] T. Jiang, J. S. Baras, "Ant-based adaptive trust evidence distribution in MANET", *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops*, pp. 588–593, 2004.

[12] A. J∅?sang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, 2007.

[13] A. P. Lauf, R. A. Peters, W. H. Robinson, "A distributed intrusion detection system for resource-constrained devices in ad-hoc networks", *Ad Hoc Networks*, vol. 8, no. 3, pp. 253–266, 2010.

[14] H. Li and M. Singhal, "Trust management in distributed systems," *IEEE Computers*, vol. 40, no.2, pp. 45–53, Feb. 2007.

[15] R. Li, J. Li, P. Liu, J. Kato, "A novel hybrid trust management framework for MANETs", in *29th IEEE International Conference on Distributed Computing Systems Workshops*, pp. 251–256, 2009.

[16] X. Li, Z. Li, P. Zhang, R. Zhang, H. Wang, "Trust based on demand multipath routing in mobile ad hoc networks", *IET Information Security*, vol. 4, no. 4, pp. 212–232, 2012.

[17] N. Marchang, R. Datta, "Light-weight trust-based routing protocol for mobile ad hoc networks", *IET Information Security*, vol. 6, no. 2, pp. 77–83, 2012.

[18] J. Martin, L. Manickam, S. Shanmugavel, "Fuzzy based trusted ad hoc on-demand distance vector routing protocol for MANET," in *Advanced on Computer Communication (ADCOM'07)*, pp. 414–421, 2007.

[19] P. Michiardi and R. Molva, "Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks", in *Communication and Multimedia Security Conference (CMS'02)*, pp. 107–121, Sept. 2002.

[20] A. Oberle, A. Rein, N. Kuntze, C. Rudolph, J. Paatero, A. Lunn, P. Racz, "Integrating trust establishment into routing protocols of today's MANETs", in *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 2369–2374, 2013.

[21] A. A. Pirzada, C. McDonald, A. Datta, "Performance comparison of trust-based reactive routing protocols," *IEEE Transactions on Mobile Computing*, vol. 5, no. 6, pp. 695–710, 2006.

[22] M. Seredynski, R. Aggoune, K. Szczypiorski, D. Khadraoui, "Performance evaluation of trust-based collaborative sanctioning in MANETs", in *12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 81–88, 2013.

[23] Bo Wang, X. Chen, W. Chang, "A light-weight trust-based QoS routing algorithm for ad hoc networks", *Pervasive and Mobile Computing*, vol. 13, pp. 164–180, 2014.
[24] X. Wang, "Intrusion detection techniques in wireless ad-hoc networks", in *IEEE 37th Annual Computer Software and Applications Conference*, pp. 347–349, 2006.
[25] H. Xia, Z. Jia, L. Ju, X. Li, E. H. M. Sha , "Impact of trust model on on-demand multi-path routing in mobile ad hoc networks", *Computer Communications*, vol. 36, no. 9, pp. 1078–1093, May 2013.
[26] H. Xia, Z. Jia, L. Ju, Y. Zhu, "Trust management model for mobile ad hoc network based on analytic hierarchy process and fuzzy theory", *IET Wireless Sensor Systems*, vol. 1, no. 4, pp. 248–266, Dec. 2011.
[27] H. Xia, Z. Jia, X. Li, L. Ju, E. H. M. Sha, "Trust prediction and trust-based source routing in mobile ad hoc networks", *Ad Hoc Networks*, vol. 11, no. 7, pp. 2096–2114, Sept. 2013.
[28] H. Xia, Z. Jia, E. H. M. Sha, "Research of trust model based on fuzzy theory in mobile ad hoc networks", *IET Information Security*, vol. 8, no. 2, pp. 88–103, Mar. 2014.
[29] M. G. Zapata, N. Asokan, "Secure ad hoc on-demand distance vector routing", *ACM Mobile Computer Communication Reviewing*, vol. 3, no. 6, pp. 106–107, 2002.

Priority Based Resource Allocation in Hybrid Network Environment

Yuvaraj Rengasamy[1], Adiline Magrica[2]
(Corresponding author: Yuvaraj Rengasamy)
Department of Computer Science and Engineering, ARS College of Engineering, Chennai, India[1]
The Detailed Postal Address of the First Author
Department of Computer Science and Engineering, Sairam College of Engineering and Technology[2]
(Email: rengasamy.yuvaraj@gmail.com)
(Received Apr. 27, 2015; revised and accepted July 28 & Aug. 19, 2015)

Abstract

This paper deals about the resource allocation hybrid cloud computing environment. Many papers deal about the resource allocation in a static environment. Some special author occasionally expressed the details of resource allocation in dynamic mobile environment. Many authors strongly explain the implementation of BigData concepts in cloud computing concept. Our proposed scheme is the combination of all. For handling of BigData we take Hadoop Frame work. In a static environment, we include the mobile nodes to handle the some of the data which is already occupied in 2 locations static network. Then prove to move that information into one dynamic wireless node for easy accessing the data. Kept the data in three location is slightly create the resemblance of Hadoop framework. The priority for resource allocation is done by on the basis of shortest length in many papers. Here we use two kinds of shortest distance using Dijkstra's algorithm shortest path algorithm for static network. Kruskal's algorithm shortest path in the dynamic mobile network. To increase the efficiency of the priority based resource allocation is the main objective of this architecture. This architecture also reduces the lagging time after fetch the request towards the server.

Keywords: BigData; Hadoop; Resource Allocation

1 Introduction

Cloud computing is the practice to use a network of remote computer (act as a server) hosted on the internet to store, manage and manipulating the data, rather than a local system. Generally, Cloud computing providers deliver supporting application through the Internet, but the business related software and logics and its corresponding information are stacked away on remote host. They are also concerned to agreed SLA, to scheduling resources in efficiently and deploying their applications on proper Virtual Machine as Objective of the SLA. They also concern the performance of the applications should be optimized manner. Now a day's most of the researchers worked on the cloud computing area and also, there exists a more work done on scheduling of applications in Clouds [4, 6, 10]. These works mainly deal about the objectives of such as cost of execution, execution time, etc. Another one research said to use the solution to Resource allocation found to be inferred [11], when the ordinary scheduling algorithm with multiple SLA objective and its corresponding SLA parameters. Most of the available solutions are based on the use of heuristics. The available solutions are based on the use of heuristics. Any job submits to cloud computing system that job generally portioned as several small tasks. Now we doubt about those tasks, are executing parallel processing or not. If it is done on parallel processing i) which order the task or executed? ii) How resource allocation task will be executed iii) when virtual machines prepare, switch the task how to schedule the overhead. Resource allocation and tax scheduling system may give the solution to these problems.

From the research work [3, 5, 8, 9] speak about the high performance computing embedded system handling task scheduling and resource allocation. Those systems worked based on two distinct steps or processes such preliminary static planning steps which is mainly concerned about grouping the set of VMs, then classify them finally deployed on physical hosts; the next step is dynamic resource providing which mainly concentrate on the add the additional resources, creation and migration of VMs due to varying workload. The preliminary steps running during initial setup and second step run continuously at production time. This paper mainly focuses both this step because initialization is the key point of this system and second step is considered as a working phase of that system. The step to working

on the basis of agreed SLA objective. Our proposed algorithm dynamically tries to respond to the colliding work load by preempting the currently executed task. In some of the cases pre-emption is not possible due to the same priority to the current and allocated task.

1) Initial static planning step: the initially group the set of VMs, then classify them and deployed onto a set of physical hosts [7];

2) Dynamic resource provisioning: the allocation of additional resources, creation and migration of VMs [1, 2], dynamically responds to varying workload. Step 2 runs continuously at production time where in contrast Step 1 is usually performed at the initial system set up time and may only be repeated for overall cleanup and maintenance on a monthly or semiannually schedule.

2 Related Work

The resource allocation application will automatically call the web application where the data are stored in data center environments. Some special application the each server has a copy of all web applications running in the system. The algorithm for dispatcher in a front-end L7-switch make the request when the number of an utilized server is below the allocated one. The network flow algorithm can allocate the load of an application among its running server. On the other hand, if we are moving to the connection oriented internet services like Chatting messenger follow an integrated approach for load dispatching and server provisioning. Amazon EC2-style environment where it places no such kind of restriction while running on the application software. Actually, it treated the VM as a black box. Most of the researchers deal only static or dynamic network. On the other words, this system will try both the network in the same system named as hybrid.

Resource allocation by live VM migration VM live migration is a widely used technique for dynamic resource allocation in a virtualized environment. Our work also belongs to this category. Sandpiper combines multi-dimensional load information into a single Volume metric. It sorts the list of PMs based on their volumes and the VMs in each PM in their volume-tosize ratio (VSR). This unfortunately abstracts away critical information needed when making the migration decision. It then considers the PMs and the VMs in the presorted order. We give a concrete example in Section 1 of the supplementary file where their algorithm selects the wrong VM to migrate away during overload and fails to mitigate the hot spot. We also compare our algorithm and theirs in real experiment. In addition, their work has no support for green computing and differs from ours in many other aspects such as load prediction.

3 Problem Definition

The main problem now facing is resource availability at lastly but not available at a require time. Now we will provide the data as a temporary manner from the mobile environment. For this purpose, we design a special frame work with the concept of Hadoop and mobile storage. The Hadoop generally handles the big data, with notation of HDFS. Along with we add one No SQL like Casandra or Hbase. The main process proposed using this framework is the total storage area is divided into two kinds one is static main storage and another one is dynamic mobile storage. The main storage is framed by Hbase and HDFS. The mobile storage is framed by android mobile storage. The main storage is the static one and mobile storage is dynamic one.

Before we knowing this system, we must know the concept of Hadoop Architecture

3.1 The Hadoop Distributed File System (HDFS)

HDFS is a fault tolerant and self-healing distributed file system designed to bend a cluster of industry standard servers into a massively scalable area of storage. Developed specifically for large size data processing workloads where flexibility, scalability and throughput are so critical, HDFS accepts data in any format of data, is optimized for high bandwidth streaming, and its scales to prove deployments of up to nearly 100 Peta Byte.

Key HDFS Features:

- Maximum Scale Architecture - To increase capacity add many hardware servers;
- High Availability - Serve mission-critical workflows and applications;
- Fault Tolerance - Automatically and seamlessly recover from failures;
- Easy Access - Multiple and open frameworks for serialization and file system mounts;
- Load Balancing - Even Place maximum data, work with its extreme efficiency and utilization;

- Tunable Replication - Many copies of each file provide information security and computer operation;
- Security - POSIX-based file permissions for users and groups with optional LDAP integration.

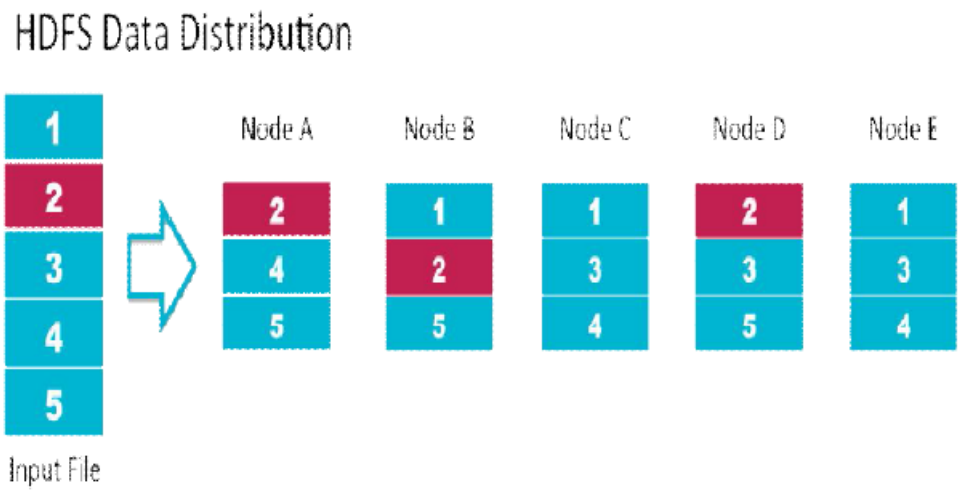

Figure 1: HDFS Data Distribution

Data in HDFS is replicated across multiple nodes for compute performance and data protection. For more information on the HDFS architecture and Figure 1.

MapReduce:

MapReduce is a massively scalable, a parallel processing framework that w Works in tandem with HDFS. With MapReduce and Hadoop, compute is executed at the location of the data, rather than moving data to the compute location; data storage and computation coexist on the same physical nodes in the cluster. MapReduce processes exceedingly large quantities of data without being met by traditional bottlenecks like network bandwidth by taking advantage of this data proximity.

Key MapReduce Features:

- Maximum Scale-out Architecture - Add servers to increase processing power;
- Security & Authentication - Works with HDFS and HBase security to prepare sure that only approved users can go against the data in the system;
- Resource Manager - Employs data locality and server resources to determine optimal computing operations;
- Optimized Scheduling - Completes jobs, according to priorities;
- Flexibility - Procedures can be composed in almost any programming language;
- Resiliency & High Availability - Multiple job and task trackers ensure that jobs fail independently and restart automatically.

MapReduce divides workloads up into multiple tasks that can be executed in parallel.

Load Balancing:

Optimization of Virtual Machine load is a process of reassigning the total load to the individual Virtual Machines to make resource utilization effective and to amend the response time of the task. A load balancing algorithm which is active in nature does not weigh the previous state or conduct of the system, that is, it depends on the present behavior of the organization. The important things to study while developing such algorithm are, estimation of load, comparison of load, performance of the Virtual Machine, nature of work to be transferred, selecting of Virtual Machine and many other ones. This lad considered can be in terms of CPU load, amount of memory used, delay or Network load.

The time needed for finishing a job inside one operation is real gamey. Thus the project is split into a number of sub projects and each sub task is passed on one task. The Proposed Load balancing algorithm is split into two stages.

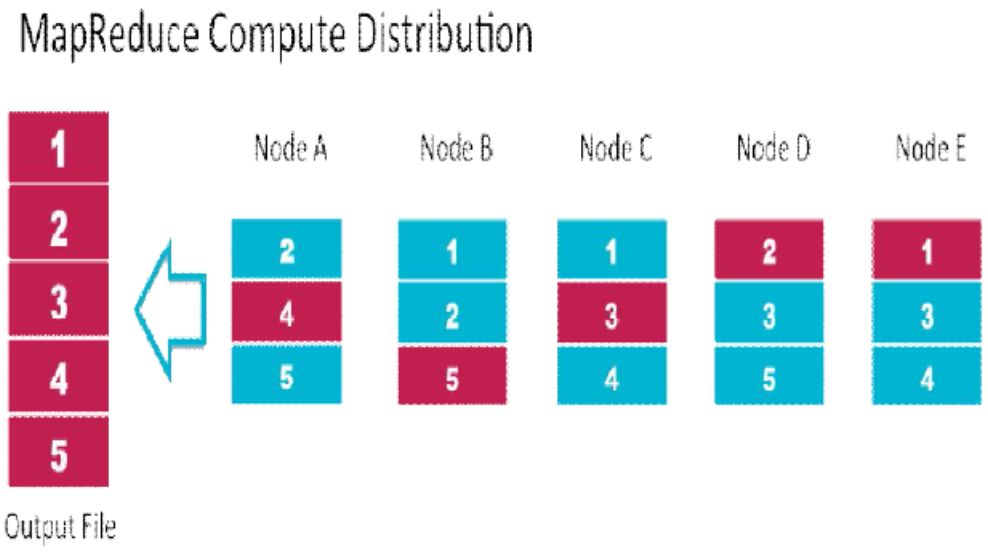

Figure 2: MapReduce Compute Distribution

A two-level task scheduling mechanism based on load balancing to meet dynamic requirements of users and obtain a high resource utilization. It achieves load balancing by first mapping tasks to Virtual Machines and then Virtual Machines to host resources, thereby improving the task response time, resource usage and overall performance of the Cloud computing environment. In the first phase, find the CPU usage and storage taken for each representative and also find available CPU cycle and memory of each VM. In a second phase compare the available resources and required resources, if resources are available instance is to be added otherwise discard the instance finally returns instance status to user. The CloudWatch monitoring service is a special storage engine that is designed for time series data. On one end data collected periodically from servers and from other services is pumped into the monitoring store, and at the other end clients can run queries against the store to extract data from it.

3.2 The Proposed Architecture

In this architecture, we have set of Hadoop hardware commodity and mobile cloud. We normally know that the Hadoop can have the 3 copies of data, Two copies are retained in the same rack. One will be on the different rack. The first two racks are same as Hadoop commodity hardware and the third is in other rack of Hadoop commodity hardware. Along with that 3 copies, this system can also take many copies of demanded data in the mobile cloud. The mobile data can easily available to the requested user rather than static storage area. If the user can give the request to the data, the load balancing system can calculate the data available, the first preference of the load balancing system is mobile data. Because it's in a dynamic environment. If it is not available the load balancing system can proceed towards the static web. The cost of the static network will be very high. Because most of the combined and stored in HDFS or HbasedB. Before storing the data, the Map Reduce module can analyze the redundancy of the data and lastly it will pull through the data in distributed environments after remove the redundancy of the information.

In Load balancing module two different projects will run short on; one is compute the shortest track of the request data address and the priority calculate module to the user. The shortest path will be calculated by using, Kruskal and Dijikstra algorithm. The algorithm selection is applied according to the selection of the data, from the cloud. After calculation of the shortest path then only the load balancer can verify the priority of the user then give the address of the data server.

3.3 Priority Based Resource Allocation in Load Balancing

The priority based resource allocation in load balancing is show in Algorithm 1.

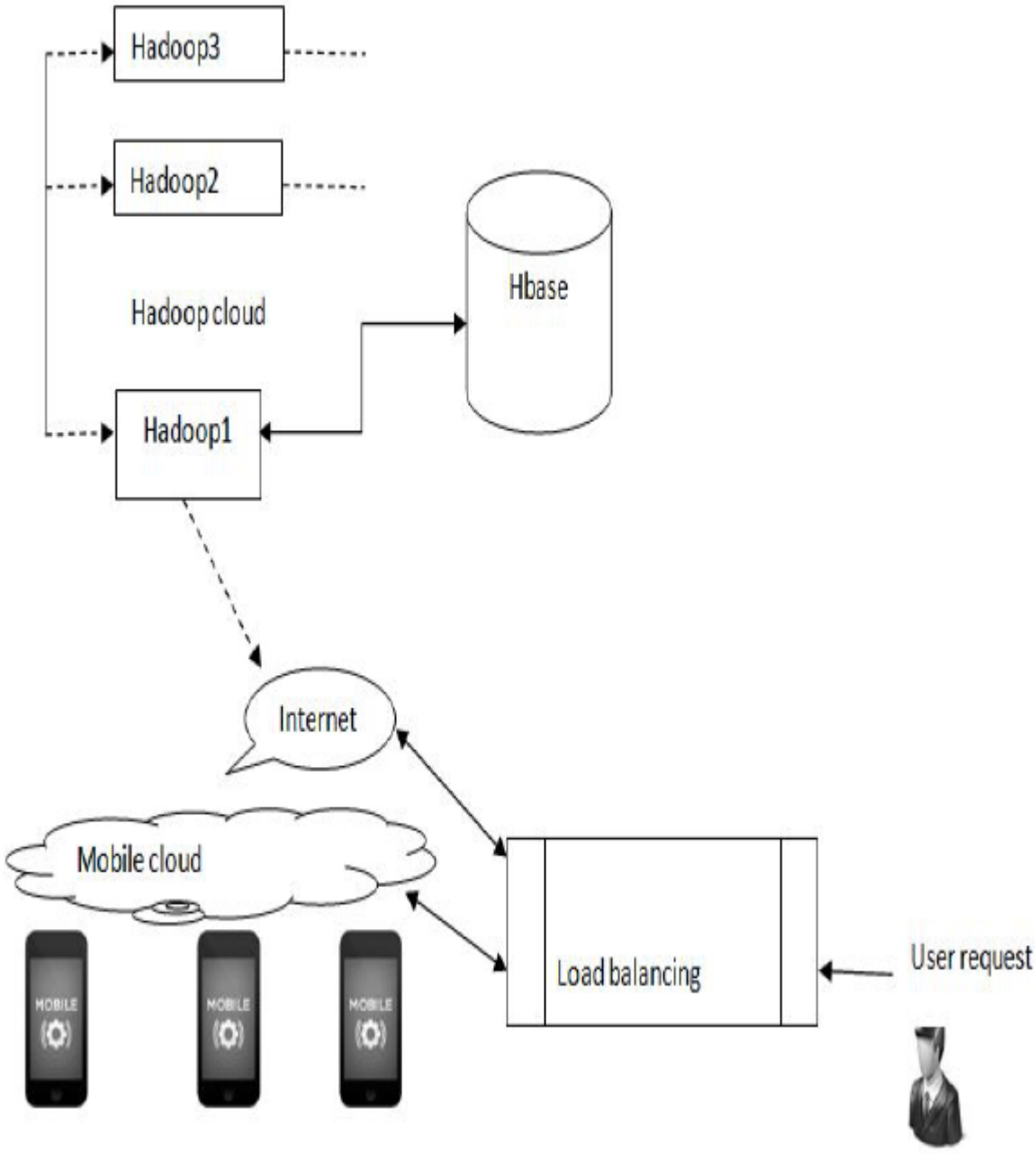

Figure 3: The Proposed Architecture

Algorithm 1 The priority based resource allocation in load balancing

```
Input: UserServiceRequest
//Call Algorithm 2 to form the list of task based on priorities;
getglobalAvailableVMList and gloableUsedVMList and also availableResourceList from each cloud schedular;
// find the appropriate VMListfromeach cloud scheduler
If AP(R, AR) ≠ φ then
// call Algorithm 1 load balancer
deployableVm=load-balancer(AP(R,AR));
Deploy service on deployableVM
deploy=true
Else if R has advance reservation and best effort task is running on any cloud then 11.
// Call Algorithm 3 CMMS for executing R with advance reservation
Deployed=true
Else if globalResourceAbleToHostExtraVM then
Start newVMInstance
Add VMToAvailbaleVMList
Deploy service on newVM
Deployed=true
Else
queueserviceReuest until
queueTime¿waitingTime
Deployed=false
End if
If deployed then
return successful
terminate
Else
return failure
terminal
```

4 Implementation

The execution of this system is acted in the Ubuntu environment. In the Ubuntu environment, we installed jdk1. 6.0, eclipse 3.4, Hadoop framework and finally Hbase. On the other side, we use the Android mobiles. All the system and mobile node connect to the common WLAN Network. The user can pass the request via, system or mobile. The data can be received from the system or mobile according to the resource availability. Most of the time it uses the mobile data only. Because the mobiles are with in the range of mobile tower with very short distance. Using suitable module to see the scope of the WLAN. About the level of the WLAN, we can see out the distance between the requested node and data node. Granting to the distance and requested time priority the load balance can easily represent the requested node and data node.

5 Results

Mobile Data:

The most of the user demanded data only movable into mobile data. Most familiar data default move to mobile environment. The un familiar data cant reach initially. They can also have the chance to move mobile environment when many user can demanded the same resource. According to the rate the data can be moved to mobile environment.

Table 1 denotes the relationship between the no. of request % in a single data and no. of % to move to mobile data. All the demanded requested not move into the mobile data. In that place our system make some weight age to that node then finally transfer the data from one static place to another mobile place as a new copy. It is clearly draw a graph in Figure 4.

Data Availability:

Table 2 denotes the relationship between the no. of request % resource availability in system node and % resource availability in mobile node. This graph show the majority on the data in mobile nodes are very familiar one. Few of them only belongs to the static nodes. It is shown in Figure 5.

Table 1: The relationship between the no. of request % in a single data and no. of % to move to mobile data

s.no	No. of request % in a single data	No. of % to move mobile data card
1	55	10
2	65	25
3	80	35
4	90	45
5	100	65

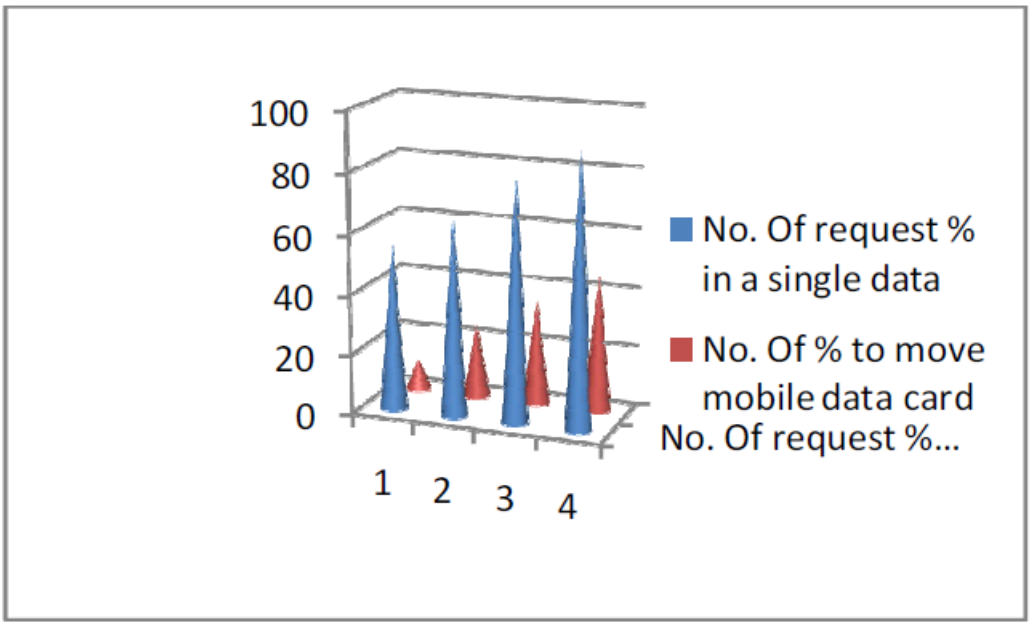

Figure 4: The Result of Moving Mobile Data

Table 2: The relationship between the no. of request % in a single data and no. of % to move to mobile data

s.no	No. of Request %	% Resource Availability in System Node	% Resource Availability in Mobile Node
1	35	0	10
2	46	20	15
3	80	30	35
4	90	45	50
5	100	65	55

Table 3: The relationship between the no. of request %, % over all data available with mobile cloud, % over all data available with out mobile cloud

s.no	No. of Request % Per Minute	% over all data available with mobile cloud	% over all data available without mobile cloud
1	35	28	25
2	46	35	30
3	80	65	55
4	90	73	45
5	100	82	60

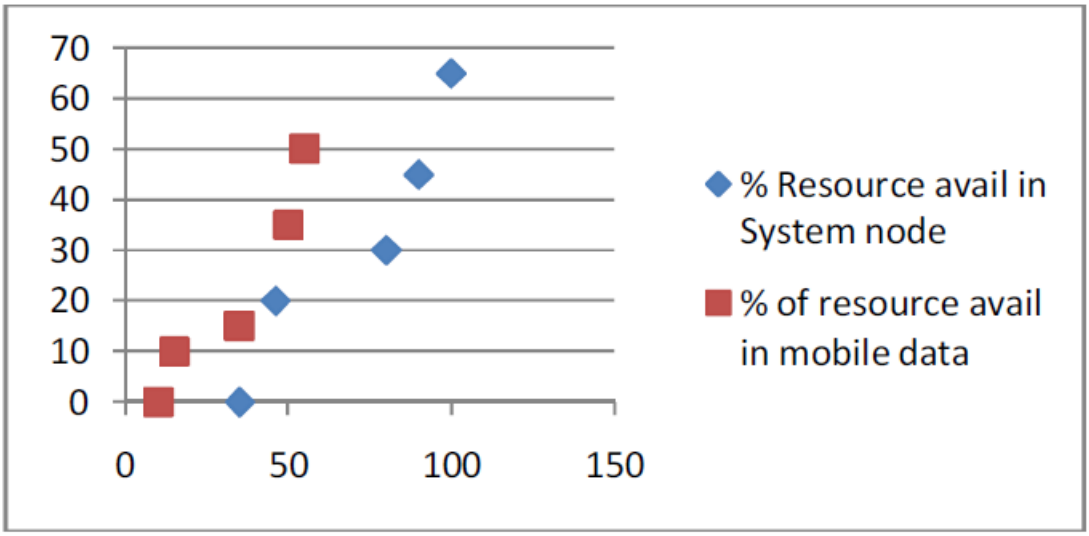

Figure 5: Resource Availability

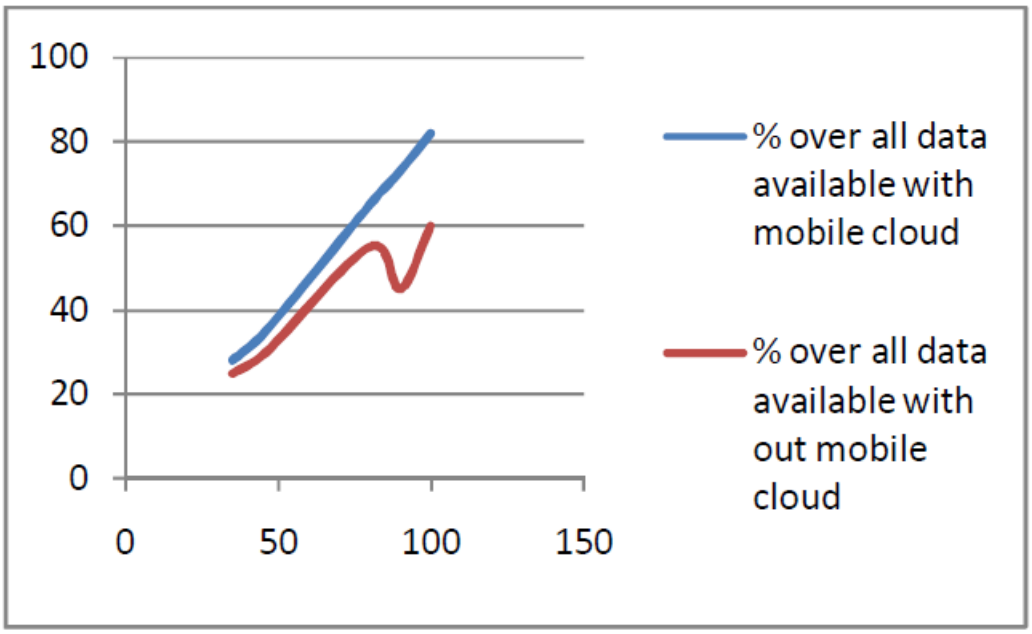

Figure 6: The Data Availability

Table 3 denotes the relationship between the no. of request %, % over all data available with mobile cloud, % over all data available with out mobile cloud. This graph show the majority of the data in mobile nodes are very familiar one. So the availability of requested data is high after completion of more number of iteration. It is shown in the graph of Figure 6.

Table 4: The relationship between the % of data in static network and % of data move in dynamic network

Days	% of data in static network	% of data move in dynamic network
1	99	1
2	98	2
3	95	5
4	92	8
5	92	8

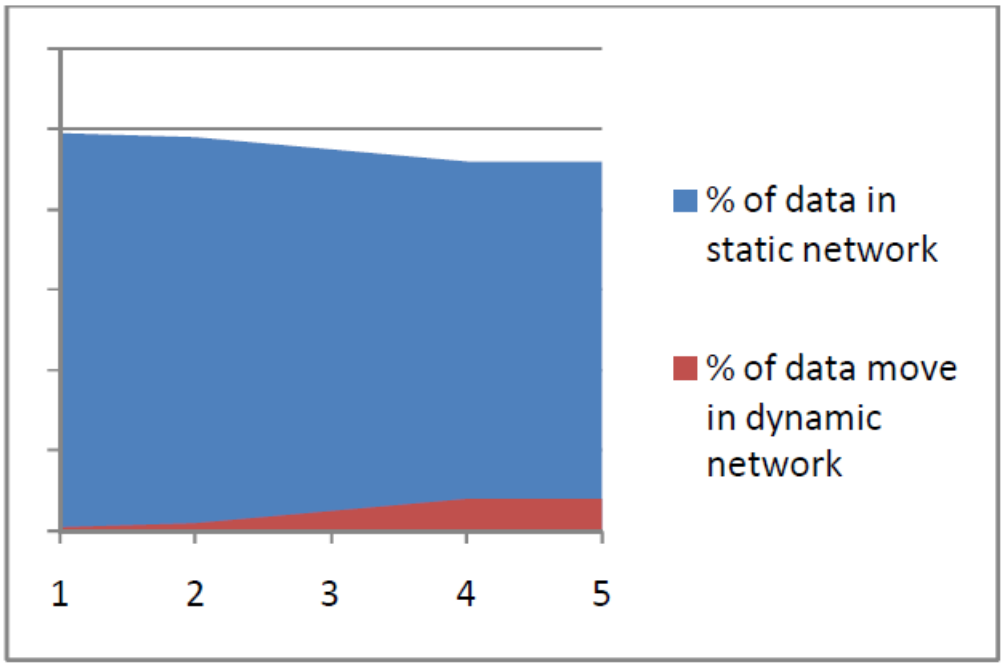

Figure 7: The Data Move in Networks

Table 4 denotes the relationship between the % of data in static network and % of data move in dynamic network. This graph show the ratio between the mobile data and static data. Actually, the static data is very huge when compared to the mobile data.

6 Conclusion

From the above result we conclude that the mobile data act a main role in resource allocation in dynamic environment. The dynamic resource allocation is a old concept in cloud computing but in mobile computing it is now developing.For the purpose it is very slow in our demo.On the following days it will move towards fast and efficient. Simply adjust the WLAN speed and load balancing capacity to improve this system very effectively and efficiently.

References

[1] M. C. Babu, S. Umamageswari, "OCRP in dynamic resource allocation using virtual machines for cloud computing environment," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 2, pp. 1002–1007, 2014.

[2] L. Dhivya, M. K. Padmaveni, "Dynamic resource allocation using virtual machines for cloud computing environment," *International Journal of Research in Engineering & Advanced Technology*, vol. 2, no. 1, pp. 1–4, 2014.
[3] A. Dogan and F. Ozguner, "Matching and scheduling algorithms for minimizing execution time and failure probability of applications in heterogeneous computing," *IEEE Transactions on Parallel and Distributed Systems*, vl. 13, no. 3, pp. 308–323, 2002.
[4] S. K. Garg, R. Buyya, and H. J. Siegel, "Time and cost tradeoff management for scheduling parallel applications on utility grids," *Future Generation Computer Systems*, vol. 26, no. 8, pp. 1344–1355, 2010.
[5] T. Hagras and J. Janecek, "A high performance, low complexity algorithm for compile-time task scheduling in heterogeneous systems," *Parallel Computing*, vol. 31, no. 7, pp. 653–670, 2005.
[6] S. Pandey, L. Wu, S. M. Guru, and R. Buyya, "A particle swarm optimization-based heuristic for scheduling, workflow applications in cloud computing environments," in *Proceedings of 24th IEEE International Conference on Advanced Information Networking and Applications (AINA'10)*, pp. 400–407, Washington, DC, USA, 2010.
[7] C. S. Pawar, B. Rajnikant, "Priority based dynamic resource allocation in cloud computing," in *IEEE International Symposium on Cloud and Services Computing (ISCOS'12)*, pp. 1–6, 2012.
[8] M. Qiu, M. Guo, M. Liu, C. J. Xue, and L. T. Yang, "Loop scheduling and bank type assignment for heterogeneous multibank memory," *Journal of Parallel and Distributed Computing*, vol. 69, no. 6, pp. 546–558, 2009.
[9] M. Qiu and E. Shaw, "Cost minimization while satisfying hard/soft timing constraints for heterogeneous embedded systems," *ACM Transactions on Design Automation of Electronic Systems*, vol. 14, no. 2, pp. 1–30, 2009.
[10] M. Salehi and R. Buyya, "Adapting marketoriented scheduling policies for cloud computing," in *Algorithms and Architectures for Parallel Processing*, LNCS 6081, pp. 351–362, Springer, 2010.
[11] J. M. Wilson, "An algorithm for the generalized assignment problem with special ordered sets," *Journal of Heuristics*, vol. 11, no. 4, pp. 337–350, 2005.

Yuvaraj Rengasamy received the MCA Degree from Anna University, Chennai in 2005 and received the Master Degree in Information Systems at University of Ballarat, Australia in 2008. He is currently working as an Assistant professor and Head of Computer Science and Engineering in Department of ARS Engineering College, Chennai. He is currently pursuing the Ph.D. degree in Computer Science and Engineering in India. His research interests include Distributed Systems, Data Mining, Mobile Adhoc Networks, next generation wireless communication systems, Cloud Computing and Big Data Analysis.

T. Adiline Magrica received her B.E. degree in Computer Science and Engineering (CSE) from the University of Madras in 1994 and received the M.E degree in CSE from the University of Madras in 2001. She got her Ph.D. degree from the Anna University, Chennai in 2011. From 2002 to till date, she is associated with Sri Sairam Engineering College, Chennai. Now, she is working as a Professor in Department of Information Technology. She has published many research papers in MANETs and member of many technical and professional societies in India. Her research interests include Mobile Adhoc Networks, next generation wireless communication systems, wireless sensor networks, Clouds and Data mining.

Efficient Computation Allocation Algorithm for Multi-View Video Coding

Hao-Wen Chi[1], Gwo-Long Li[2], Mei-Juan Chen[1] and Jie-Ru Lin[1]
(Corresponding author: Mei-Juan Chen)
Department of Electrical Engineering, National Dong Hwa University[1]
No. 1, Sec. 2, Da Hsueh Rd. Shoufeng, Hualien 97401, Taiwan, R.O.C.
(Email: cmj@mail.ndhu.edu.tw)
Novatek Microelectronics Corp.[2]
Hsinchu, Taiwan, R.O.C.
(Received Sept. 16, 2015; revised and accepted Oct. 20, 2015)

Abstract

In this paper, a computation allocation algorithm for multi-view video coding is proposed to aim at higher compression efficiency with considering computational resource availability. Based on analyzing results of inter-view motion vector correlation and rate distortion costs between views, the limited computation resources can be allocated efficiently according to the motion vector difference and rate distortion cost of the macroblock. In addition, in order to receive accurate global disparity vectors, a global disparity vector (GDV) normalization algorithm is also applied to smooth the global disparity vector for the purpose of shifting the information of the main view to the auxiliary view. Simulation results demonstrate that our proposed algorithm can maintain a quality performance and restrain the bit-rate increasing with low and limited computation resource support.

Keywords: Computation Allocation; Motion Vector Difference; Multi-view Video Coding; Rate-distortion Cost

1 Introduction

In order to provide the ability of compressing video data from multiple views, the multi-view video coding (MVC) [1], which is the extension of the H.264/AVC video coding standard, has been standardized. The coding blocks of MVC are almost the same as the intrinsic coding blocks of H.264/AVC except that the disparity estimation operations are also executed in different views. Figure 1(a) and Figure 1(b) illustrate the frame referring structures between frames and views for H.264/AVC and H.264/MVC [2], respectively. As a result, the computational complexity of H.264/MVC is expected to be much higher than that of H.264/AVC since additional disparity estimation operations are required between views.

Lowering the computational burdens of H.264/MVC with slightly sacrificing PSNR and increasing ignorable bit-rate receive many attentions in the research of H.264/MVC. Shen et al. [3] and Peng et al. [4] use the mode and rate distortion cost information from previous frames and views to predict the best mode of the current macroblock. Zhu et al. [5] observe that disparity vectors (DV) always have a disorder phenomenon by analyzing the disparity vector correlation between views. To solve this problem, the authors present some equations to smooth the disparity vectors with large variations. Afterwards, the smoothed disparity vectors can be used as the search center for the current block by incorporating a search range decision algorithm. The search range algorithm proposed in [5] further considers the disparity vector variation in the temporal domain. Also some researches [6, 7, 8] have discussed how to match the correlation between views by moving the global disparity vectors.

Motivated from the high computational complexity concern of H.264/MVC, this paper proposes a computation allocation algorithm for multi-view video coding to reduce the computational overheads and effectively use the computational resources. Based on inter-view motion vector analysis [9], the computational resource allocation algorithm is thus proposed to effectively dispatch the computation for mode decision and motion estimation.

The remainder of this paper is organized as follows. Section 2 describes the analysis of motion vector difference and rate distortion cost of different views in MVC. The proposed algorithm is presented in Section 3. Section 4 shows the simulation results. Finally, Section 5 gives the conclusion.

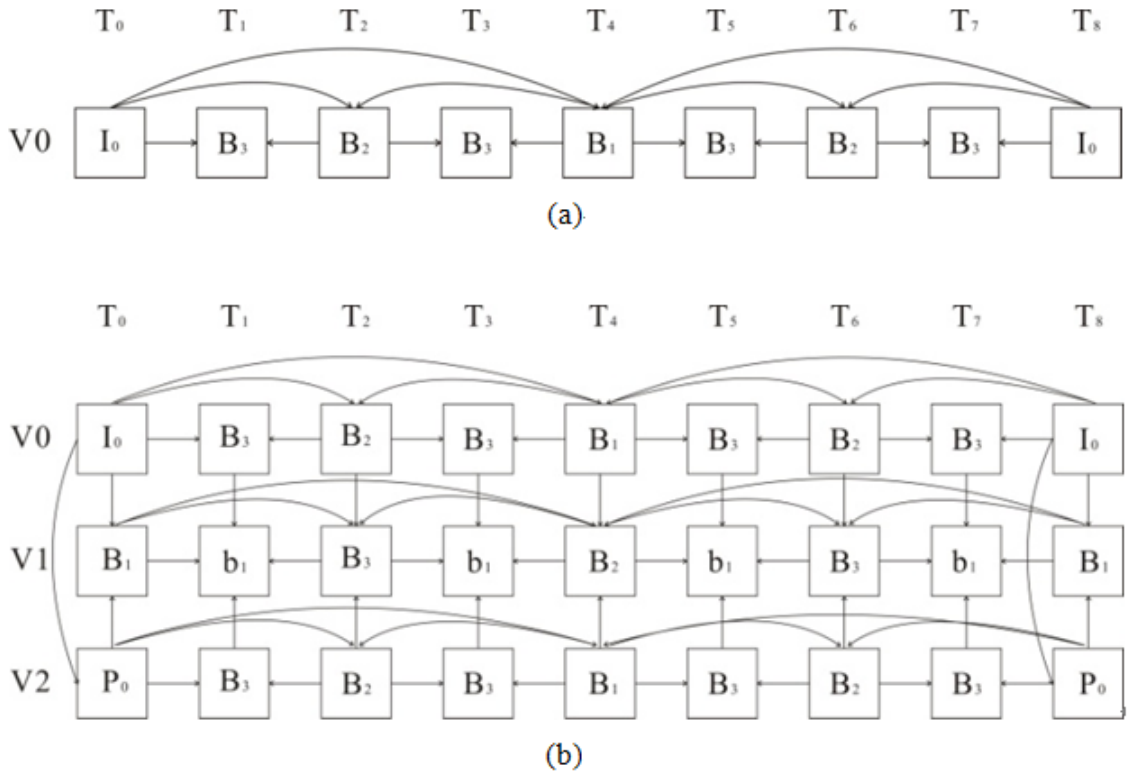

Figure 1: Reference frame structure (a) H.264/AVC (b) H.264/MVC

2 Analysis of MVD and RD-cost

2.1 Global Disparity Vector Normalization

To accurately analyze the relationship between views, the effect caused by inaccurate global disparity estimation as mentioned in [5] has to be eliminated. Therefore, we modify some equations mentioned in [5] to further improve the accuracy of the global disparity vector. In our modification, the disparity vector, $DV(i,j)_t$, of macroblock (i,j) at current frame t, and the normalized global disparity vector $NGDV_{t-1}$ at the previous frame is used to calculate the angles. If the angle between disparity vector of the current encoding macroblock and the previous global disparity vector $NGDV_{t-1}$ exceeds 90^o, the disparity vector will require to be adjusted. The normalized global disparity vector can be calculated as Equation (1):

$$NGDV_t = \left\{ \begin{array}{ll} \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} NDV(i,j)_t}{M\times N} & |t>0 \\ \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} DV(i,j)_t}{M\times N} & |t=0 \end{array} \right\} \quad (1)$$

where t is the frame index; M is the number of macroblocks in the frame width; N refers to the number of macroblocks in the frame height; and $NDV(i,j)_t$ means the normalized disparity vector of macroblock(i,j) at current frame t. The calculation of $NDV(i,j)_t$ in [5] is listed in Equation (2), (3), (4) as follows:

$$NDV(i,j)_t = median(DV'(i,j)_t, DV(T)_t, DV(B)_t, DV(L)_t, DV(R)_t) \quad (2)$$

$$DV'(i,j)_t = \left\{ \begin{array}{ll} DV(i,j)_t, & \text{If } \theta(DV(i,j)_t, NGDV_{t-1}) \le 90^o \\ DV_{PRED}, & \text{Otherwise} \end{array} \right\} \quad (3)$$

$$DV_{PRED} = \frac{\sum DV(NB)_t}{K}, \quad \text{where} \quad DV(NB)_t \in R \quad (4)$$

$$R = \left\{ DV(NB)_t \left| \begin{array}{l} \theta(DV(NB)_t, NGDV_{t-1}) \le 90^o \\ |DV(NB)_t^x - DV(i,j)_t^x| + |DV(NB)_t^y - DV(i,j)_t^y| = d_{min} \end{array} \right. \right\} \quad (5)$$

where $DV(T)_t$, $DV(B)_t$, $DV(L)_t$, and $DV(R)_t$ indicate the neighboring disparity vectors as shown in Figure 2. In Equation (5), x and y denote the magnitude of the disparity vectors in horizontal and vertical directions, respectively.

K is the size of set R. $\theta(a,b)$ is the function which can be used to calculate the angle between element a and b. If the angles between $DV(NB)_t$ and $NGDV_{t-1}$ are less than 90^o, we propose to calculate the minimum Manhattan distance between $DV(i,j)_t$ and $DV(NB)_t$. After the above normalization process, we can get smoother disparity vectors as shown in Figure 3.

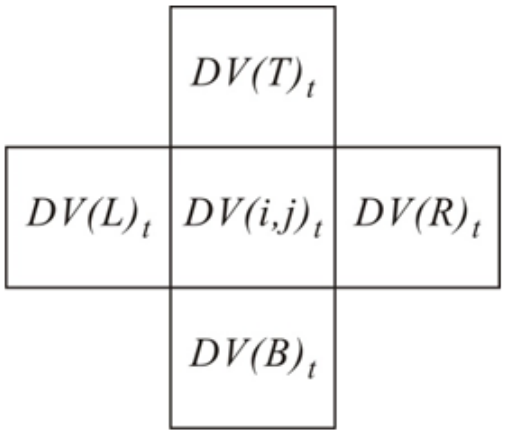

Figure 2: The distribution of disparity vectors and neighboring disparity vectors (DV_{NB})

Figure 3: A field of disparity vectors for the 25th frame of V1 in Ballroom sequence (a) original disparity vector (b) normalized $DV(i,j)_t$

2.2 Motion Vector Difference Analysis

In this subsection, we provide some analyses to show the motion vector relationship between views. The analytical conditions are described as follows. The test sequences of Ballroom and Exit are used; Search Range (SR) is set to ± 96 pixels; two reference views are used; and Group Of Picture (GOP) is 16.

Figure 4 shows the maximum absolute motion vector difference relationship for three different views. Here, the maximum absolute motion vector difference for macroblock (i,j) is calculated by Equation (6) in which the terms of $MVD(i,j)^x$ and $MVD(i,j)^y$ are the motion vector differences in horizontal and vertical directions, respectively.

$$MVD(i,j) = MAX(|MVD(i,j)^x|, |MVD(i,j)^y|). \tag{6}$$

From Figure 4, we can observe that the maximum absolute motion vector difference has slight disparity between views due to the camera setting position in multi-view system. Therefore, it means that it is possible to predict the

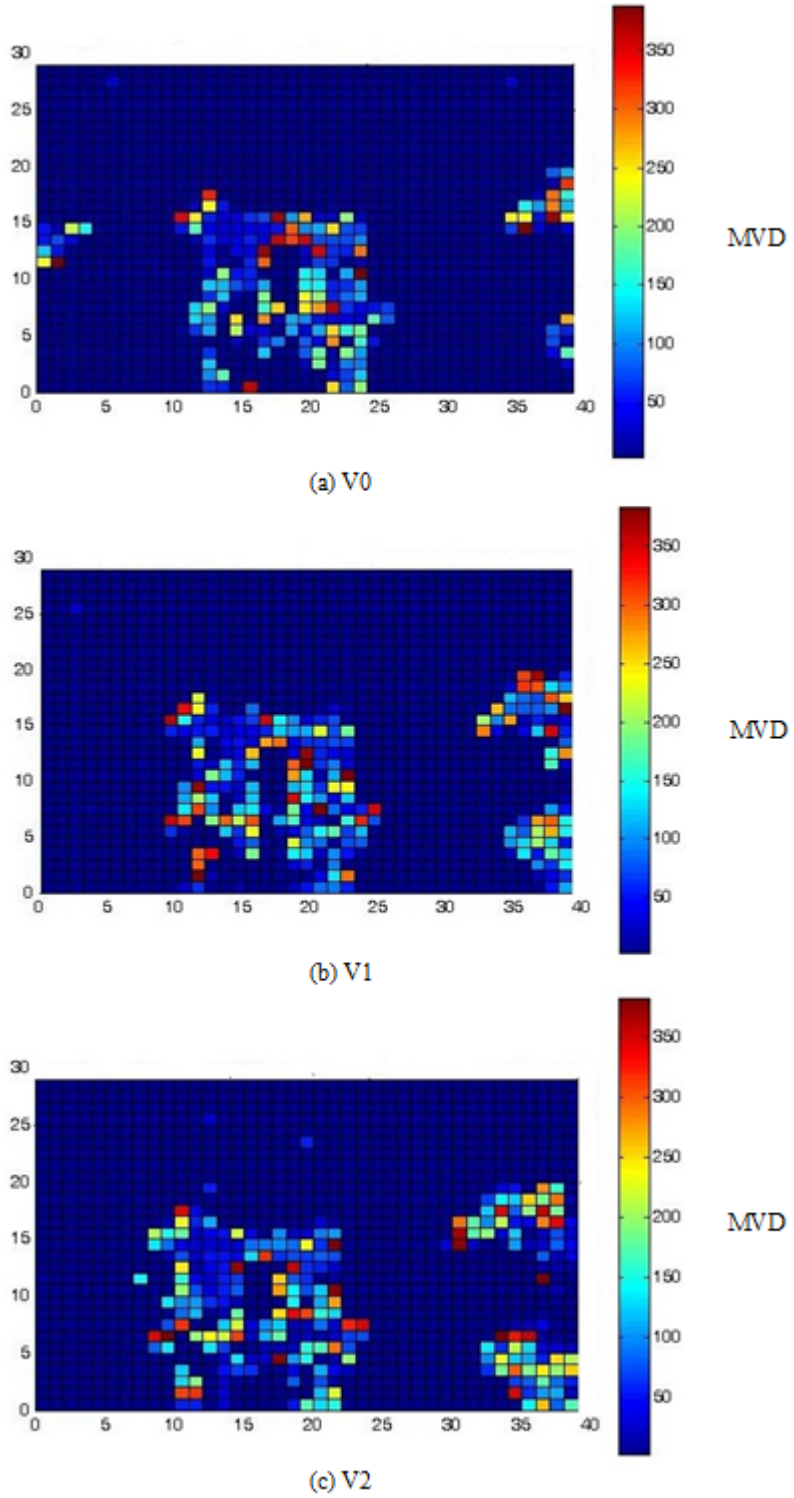

Figure 4: Motion vector difference in the temporal domain of different views for Ballroom (640 × 480) sequence (a) V0 (b) V1 (c) V2

motion vector difference of a certain view from another view by simply considering the disparity. Figure 5 illustrates how the motion vector differences are moved by considering disparity. As a result, we further analyze the correlation for motion vector difference between the main view and the auxiliary view by using $NGDV_{V0}$ and $NGDV_{V2}$. Let D_{V0} denotes the disparity which comes from V0, D_{V2} denotes the disparity which comes from V2, which are defined in Equation (7). Here, the difference between motion vector differences is denoted as $DMVD$ for V0 and V2 as Equation (8) shown.

$$\begin{aligned} D^x_{V0} &= \left[\frac{NGDV^x_{V0}}{16}\right] \\ D^y_{V0} &= \left[\frac{NGDV^y_{V0}}{16}\right] \\ D^x_{V2} &= \left[\frac{NGDV^x_{V2}}{16}\right] \\ D^y_{V2} &= \left[\frac{NGDV^y_{V2}}{16}\right] \end{aligned} \tag{7}$$

$$\begin{aligned} DMVD_{V0} &= MVD_{V1}(i,j) - MVD_{V0}(i + D^x_{V0}, j + D^y_{V0}) \\ DMVD_{V2} &= MVD_{V1}(i,j) - MVD_{V2}(i + D^x_{V2}, j + D^y_{V2}) \end{aligned} \tag{8}$$

Figure 6 provides the analytical results of DMVD for different test sequences. From this figure, we can observe that over 75% of macroblocks have zero DMVD. In other words, it means that the motion vector difference in the main view after shifting will be similar to the motion vector difference in the auxiliary view. In addition, the DMVD of the remaining 15% macroblocks is between the range of -10 and 10. This situation is mainly caused by the light refraction problem of the camera. Therefore, based on the analytical results shown in Figure 6, we can use the motion vector differences in the main views to predict the motion vector difference of the un-encoded macroblocks in the auxiliary view.

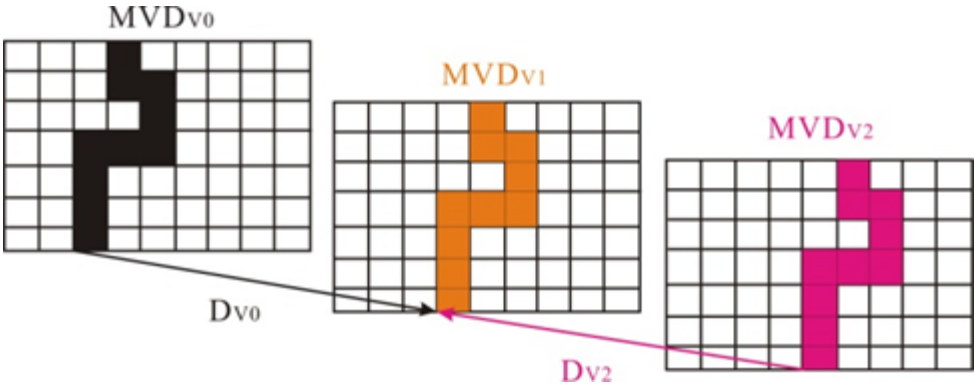

Figure 5: The motion vector differences are moved by considering disparity

2.3 Rate Distortion Cost Analysis

To achieve better rate distortion performance, the rate distortion cost values are widely adopted to evaluate the prediction results [10]. Therefore, we also analyze the rate distortion cost correlation between the main view and the auxiliary view. Figure 7 shows the analytical results of rate distortion costs. In this figure, the term J means the rate distortion cost. From this figure, we can observe that the macroblocks with higher motion activity will have higher rate distortion cost. Therefore, these analytical results can provide some information for guiding the computational complexity distribution process in our proposed algorithm.

3 Proposed Computation-Aware Algorithms

3.1 Computation Allocation in Each Frame

Figure 8 shows the flowchart of our proposed computation allocation algorithm for H.264/MVC. Our proposed algorithm is briefly described below. First, the motion vector differences and global disparity vectors of V0 and V2

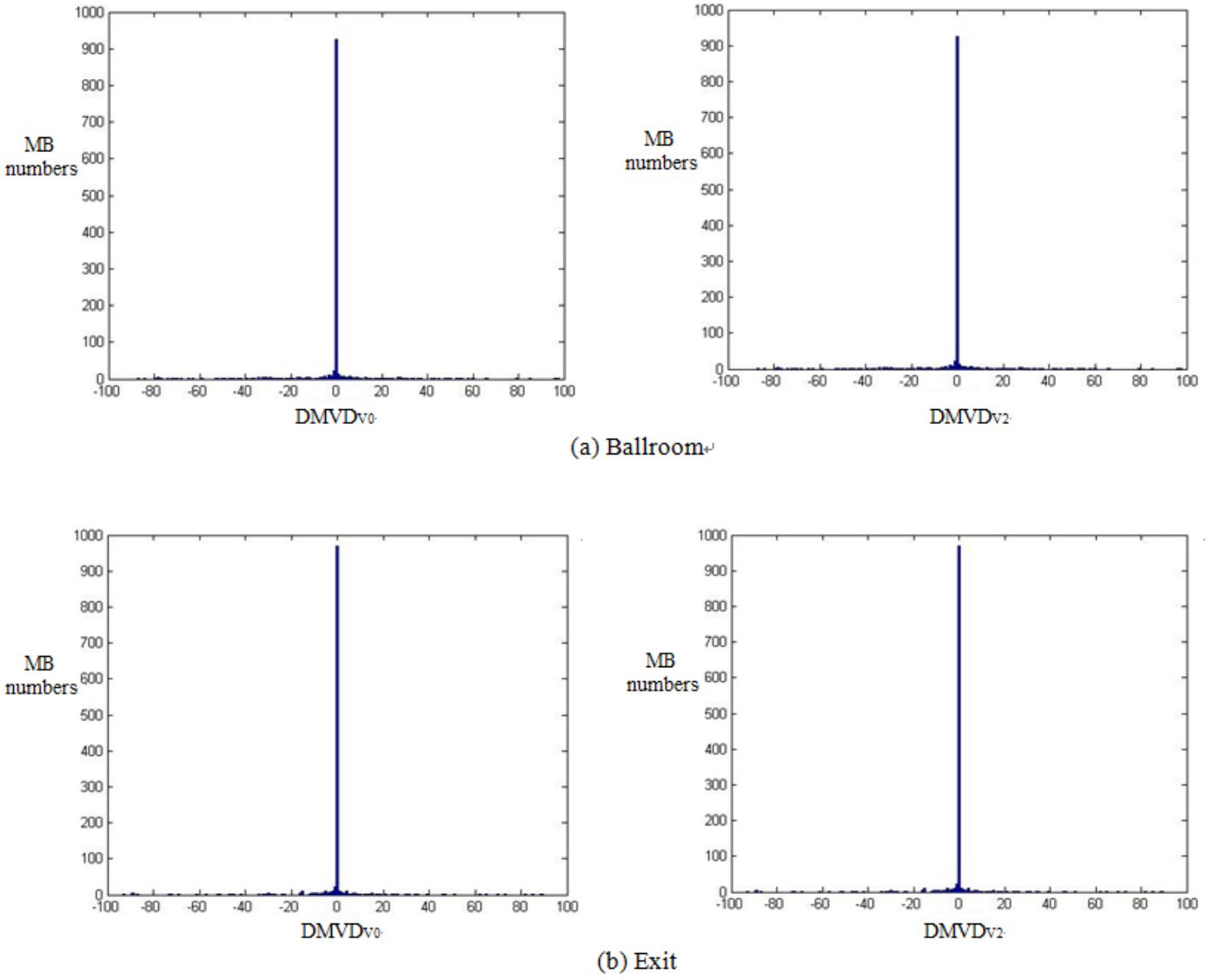

Figure 6: Histogram of differential motion vector difference(DMVD) (a)Ballroom(b)Exit sequences

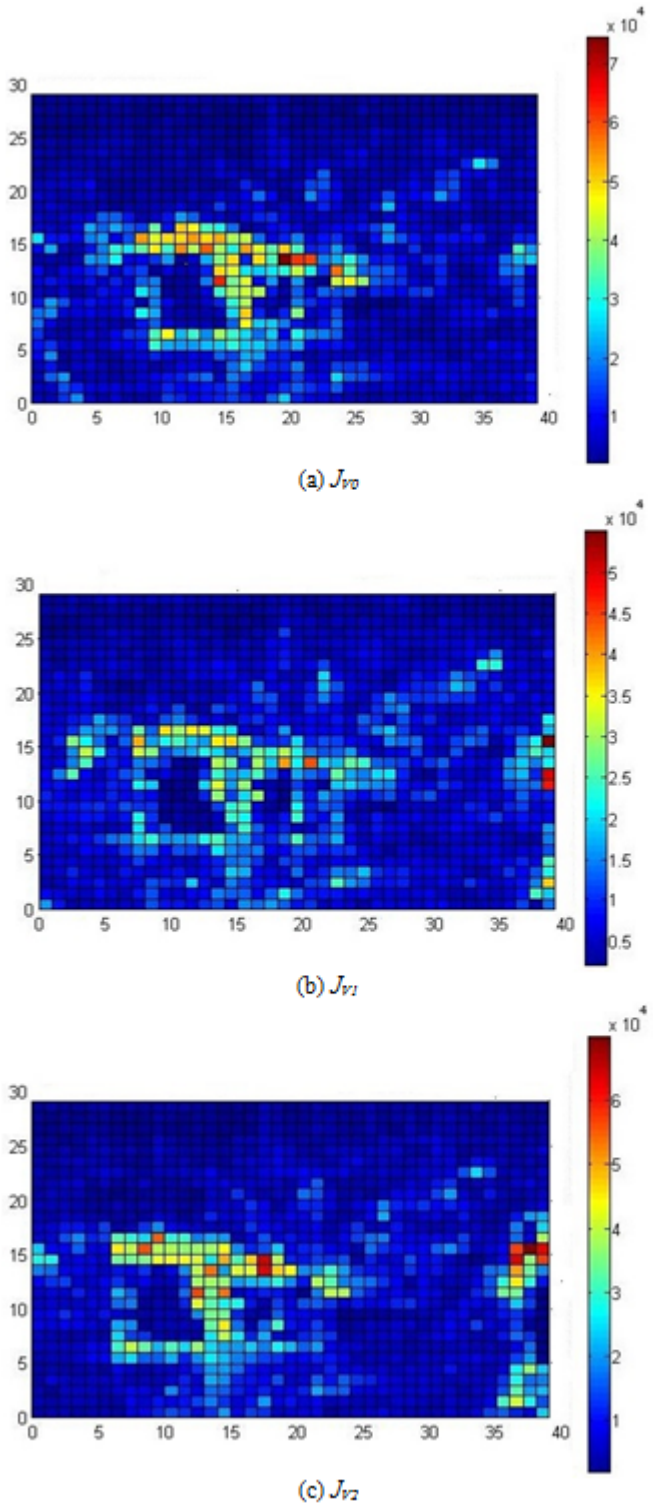

Figure 7: Rate distortion cost distribution of 16 × 16 on Ballroom(640 × 480) sequence for various views (a) V0 (b) V1 (c) V2

are obtained. Afterwards, we use the motion vector differences, which have been shifted by global disparity vectors in the main view, to predict motion vector difference value ($PMVD_{V1}$) in the auxiliary view as defined in Equation (9) and Equation (10). If the macroblocks of V1 have been covered by both of V0 and V2, the MVDs of V2 have the higher priority to be used.

$$PMVD_{V1}(i,j) = MVD_{V0}(i + D^x_{V0}, j + D^y_{V0}) \quad (9)$$

$$PMVD_{V1}(i,j) = MVD_{V2}(i + D^x_{V2}, j + D^y_{V2}) \quad (10)$$

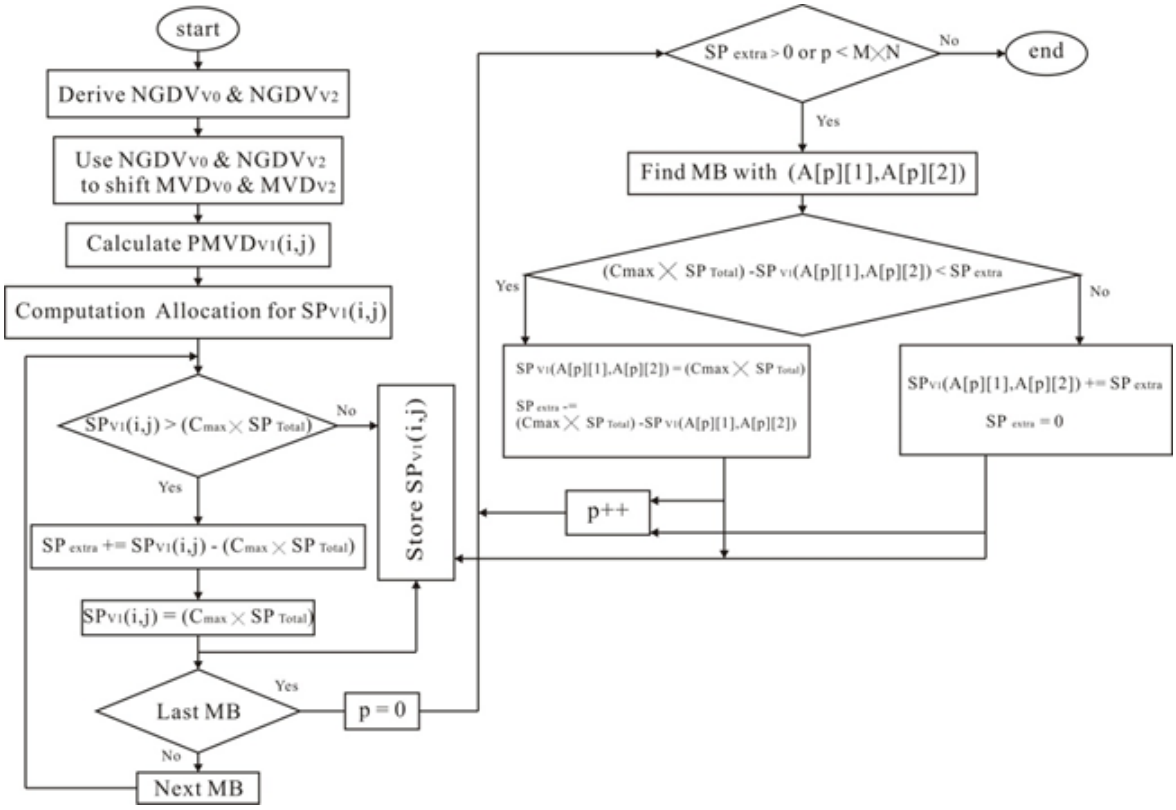

Figure 8: Computation allocation flowchart in one frame

Once the $PMVD_{V1}(i,j)$ has been calculated, it will be used for calculating the search points in the auxiliary view by using Equation (11) and the available number of search points will be calculated as well by additionally considering the computational cost as shown in Equation (12) and Equation (13).

$$SP_{Pred}(i,j) = (2 \times PMVD_{V1}(i,j) + 1)^2 \quad (11)$$

$$SP_{Total} = (2 \times SR + 1)^2 \quad (12)$$

$$SP_{V1}(i,j) = \frac{SP_{Pred}(i,j)}{\sum_{i=0}^{N-1}\sum_{j=0}^{M-1} SP_{Pred}(i,j)} \times C_{Max} \times M \times N \times SP_{Total} \times \rho \quad (13)$$

where SP_{Total} is the required search points to calculate the RD-cost of one unit of $CMode_m$. SR represents the search range. ρ can be adjusted for various coding complexities. The definition of maximum computation cost ($C_{MAX} = 7$) is based on reference paper [10] and the proportions of each mode are shown in Table 1. After the search points are allocated, the mode checking priority suggested in [11] is adopted in our proposed algorithm to execute the motion estimation process subject to the allocated search points. In [11], the mode checking priority is decided by considering the rate distortion costs as shown in Equation (14) and Equation (15).

$$Mode = \{Mode_m | 1 \le m \le 4\}, \text{ where } Mode_m \in \{mode_{16\times16}, mode_{16\times8}, mode_{8\times16}, mode_{8\times8}\} \quad (14)$$

$$J_{Mode_m}(i,j) = \{J_{Mode_m} \le J_{Mode_{m+1}} |, 1 \le m < 4\}, \quad (15)$$

where m represents index of mode. However, it has to be pointed out that since displacement exists between views, the mode checking order also needs to move to the corresponding macroblock position by considering the global

disparity as shown in Equation (16) and Equation (17).

$$J_{Mode_{m_{V1}}}(i,j) = J_{Mode_{m_{V0}}}(i+D^x_{V0}, j+D^y_{V0}) \quad (16)$$

$$J_{Mode_{m_{V1}}}(i,j) = J_{Mode_{m_{V2}}}(i+D^x_{V2}, j+D^y_{V2}), \quad (17)$$

where $Mode_{m_{V1}}$ is the predicted mode in compliance with checking order in $V1$. Here, the $m = 1$ represents the best prediction coding mode of current macroblock. Afterwards, we further define a set called A as defined in Equation (18) which consists of the information of rate distortion cost (A[p][0]) and macroblock horizontal index (A[p][1]) of all macroblocks of an entire frame and macroblock vertical index (A[p][2]) of all macroblocks of an entire frame. Similarly, the set A has to be sorted in descending order according to the rate distortion costs so that the condition of Equation (19) can be satisfied. It has to be noticed that the macroblock index stored in set A is required to be moved accordingly. Finally, the computation allocation process will be executed and repeated for macroblocks in compliance with the checking order specified in set A until the remaining search points become zero.

$$A[p] = \{\{J_{Mode_{1_{V1}}}(i,j), i, j\} | 0 \le i < M, 0 \le j < N\} \quad (18)$$

$$A[p][0] \ge A[p+1][0] \forall 0 \le p < M \times N \quad (19)$$

Table 1: Relative ration of computation time for each mode [10]

Weight	Mode					
	Direct	16×16	16×8	8×16	8×8	C_{MAX}
$CMode_m$	0	1	1	1	4	7

3.2 Computation Allocation in Each MB

Figure 9 shows the flowchart of our proposed search method algorithm. In our proposed search method, the allocated search point $SP_{V1}(i,j)$ is used to determine whether the remaining search points are able to complete a full search or not [12]. If the search points are not able to complete a full search, the search method will switch to TZ search [13] to save the computational cost.

3.3 Calibration of Predicted Motion Vector Difference

However, we can't guarantee to obtain $PMVD_{V1}(i,j)$ successfully for all macroblocks by only V0 or V2 due to the different directions and the different magnitudes of the global disparity vector. As a result, we use both MVDs of V0 and V2 to obtain $PMVD_{V1}$ for those macroblocks which lack of information as Equation (10). Figure 10 shows this condition and illustrates how the $PMVD_{V1}$ have been selected. From Figure 10, we can observe that the MVDs of V0 can be used to predict MVDs of V1 labelled by pink area. The red area is lack of MVDs, so we also use MVDs of V2 to predict MVDs of V1.

As a result, we also try different mechanisms to compare the performance:

Mechanism 1: If the macroblocks of V1 have been covered by both of V0 and V2, the MVDs of V0 have the higher priority to be used. In other words, only the unfilled area in V1 would be filled by MVDs of V2.

Mechanism 2: In this mechanism, the PMVDs of overlapped area are derived by averaging the MVDs of V0 and V2.

4 Simulation Results

This section exhibits simulation results to demonstrate the efficiency of our proposed computation allocation algorithm. The simulation settings are shown in Table 2.

The Bjontegaard delta peak signal to noise ratio (BDPSNR), Bjontegaard delta bit-rate (BDBR) [14] and coding complexity are used for comparison in our paper. Tables 3 $\sim$ 6 show the rate distortion performance and coding complexity comparisons for our proposal compared with H.264/MVC references software JMVC8.5 [2] and the graphical results are shown in Figure 11. Since our proposal could be the first work which has considered the computational resource for mode decision in H.264/MVC, we only compare our proposal to a simple method called

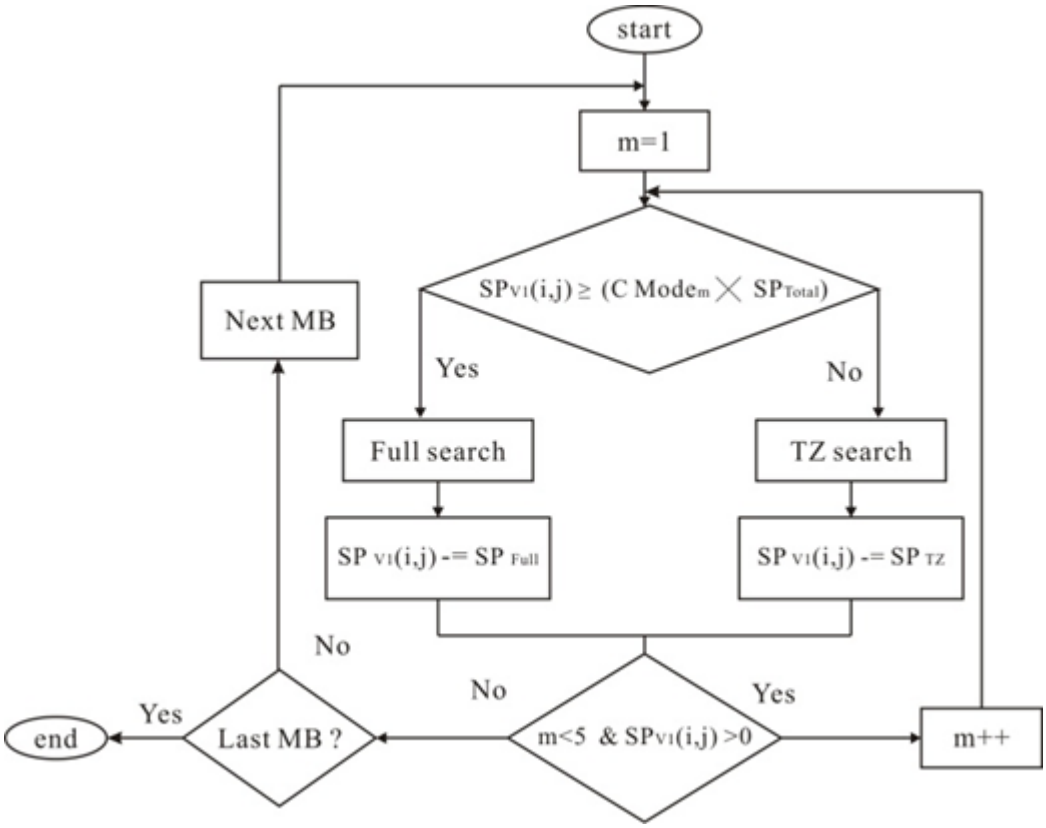

Figure 9: Computation allocation flowchart in each MB

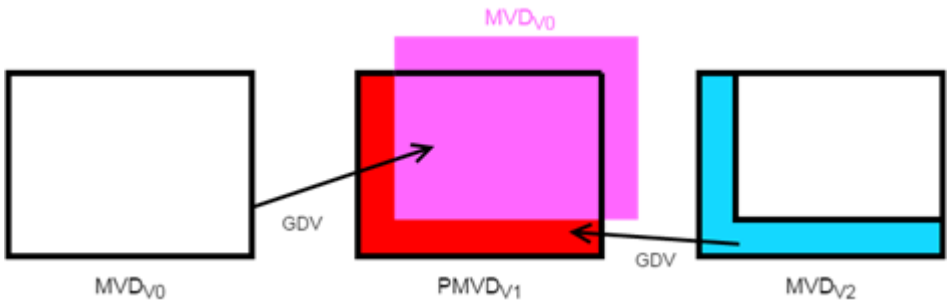

Figure 10: Illustration of PMVD selection from MVDs of V0 and V2

Table 2: Experimental environment in proposed algorithm

Codec	JMVC8.5
Test sequences	Ballroom (640 × 480), Exit (640 × 480), Race (640 × 480)
QP	28, 32, 36, 40
Search range	32
GOP	16
Encoding format	Hierarchical B
Coding frame	81

computation restricted search (CRS) method. The CRS method only evenly distributes the available search points for each prediction mode. In addition, CRS plus mode priority is to incorporate mode priority of Equation (15) to give the search points to the MB with higher priority. Afterwards, the motion estimation process by spiral search will be terminated if the allocated search points have run out. From the results, we can observe that our proposed method has better performance in both rate distortion performance and coding complexity saving. On average, our proposed algorithm can achieve 0.335dB BDPSNR improvement and a 7.4% BDBit-Rate decrease on average when compared to CRS for 640×480 resolution sequences in lower computation cost. It can achieve 0.058 dB BDPSNR improvement and a 1.1% BDBit-Rate decrease on average when compared to CRS for 1024×768 resolution sequence in lower computation cost. When we get higher computation cost for coding, our proposed algorithm can achieve 0.172dB BDPSNR improvement and a 3.7% BDBit-Rate decrease on average for 640×480 resolution sequences; our proposed algorithm also can achieve 0.040dB BDPSNR improvement and a 0.8% BDBit-Rate decrease on average for 1024×768 resolution sequences. For the coding complexity for low resolution sequences comparison, our proposed algorithm only needs 16.8% coding complexity which is much less than that of CRS. In addition, for high motion sequence, our proposed algorithm can achieve much higher performance than CRS.

Table 3: Performance comparisons of the algorithm with CRS by JMVC 8.5 (Lower computation cost)

Sequence(640×480)	BDPSNR (dB)		BDBit-Rate (kbits/s)		Coding Complexity (%)	
	CRS	Proposed	CRS	Proposed	CRS	Proposed
Ballroom	-0.319	-0.158	6.8%	2.8%	26.4	22.5
Exit	-0.232	-0.102	7.2%	3.5%	26.1	12.3
Race	-0.852	-0.165	17.7%	3.5%	21.6	15.7
Average	-0.476	-0.141	10.6%	3.2%	24.7	16.8

Table 4: Performance comparisons of the algorithm with CRS by JMVC 8.5 (Lower computation cost)

Sequence (1024×768)	BDPSNR (dB)		BDBit-Rate (kbits/s)		Coding Complexity (%)	
	CRS	Proposed	CRS	Proposed	CRS	Proposed
Ballroom	-0.149	-0.033	2.9%	0.8%	26.2	25.1
Exit	-0.067	-0.040	1.5%	0.9%	26.4	26.5
Race	-0.053	-0.022	1.1%	0.5%	26.3	26.4
Average	-0.090	-0.032	1.8%	0.7%	26.3	26.0

Table 5: Performance comparisons of the algorithm with CRS by JMVC 8.5 (Middle computation cost)

Sequence (640×480)	BDPSNR (dB)		BDBit-Rate (kbits/s)		Coding Complexity (%)	
	CRS	Proposed	CRS	Proposed	CRS	Proposed
Ballroom	-0.142	-0.066	3.2%	1.5%	50.5%	36.9
Exit	-0.095	-0.042	2.6%	1.1%	50.8%	33.4
Race	-0.525	-0.140	10.7%	2.7%	38.8%	30.0
Average	-0.254	-0.082	5.5%	1.8%	46.7%	33.4

Figure 12 shows the subjective quality and PSNR comparison of the Ballroom sequence under different computations. We can find that when the computation is 0.05%, there are slightly distortions on face of the dancer. But when the computation is up to 6%, the subjective quality and the PSNR are improved. Figure 13 shows the hit-rate which means the probability the proposed algorithm selects the same mode as JMVC 8.5 under different coding times (computations). Exit sequence has outstanding hit-rate performance; if the computation is up to 0.05%, the hit-rate is over 90%. Our proposed algorithm can accurately select the best prediction mode.

Tables $7 \sim 8$ show the performance comparison of the proposed algorithm and CRS plus mode priority for Race1 sequence. Although CRS plus mode priority gets better performance compared with CRS, our proposed algorithm can achieve 0.04 BDPSNR improvement and a 0.65% BDBit-Rate decrease compared with CRS plus priority. Table 9 shows the performance comparison of different PMVD derivation mechanisms. From this table, we can find that

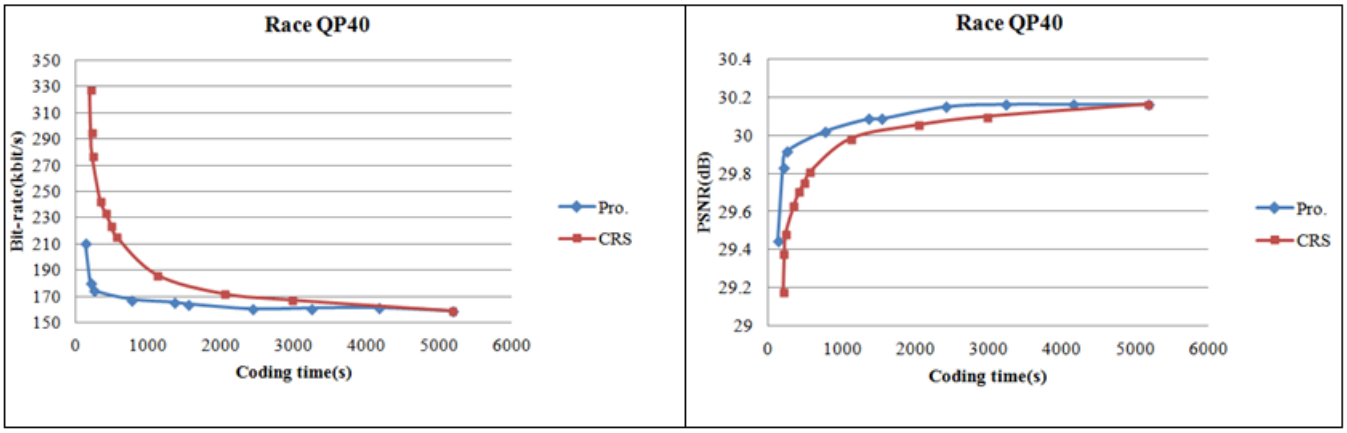

Figure 11: Performance comparison of rate-distortion curves of the proposed algorithm and CRS in V1

Figure 12: The subjective quality and PSNR comparison of the Ballroom sequence under different computations (a) 0.05% (b) 6% (c) 10% (d) 100%

Table 6: Performance comparisons of the algorithm with CRS by JMVC 8.5 (Higher computation cost)

Sequence (1024 × 768)	BDPSNR (dB)		BDBit-Rate (kbits/s)		Coding Complexity (%)	
	CRS	Proposed	CRS	Proposed	CRS	Proposed
Ballroom	-0.089	-0.006	1.7%	0.1%	50.7	48.6
Exit	-0.034	-0.023	0.8%	0.5%	51.2	50.9
Race	-0.034	-0.009	0.7%	0.2%	53.7	53.6
Average	-0.052	-0.012	1.1%	0.3%	51.9	51.0

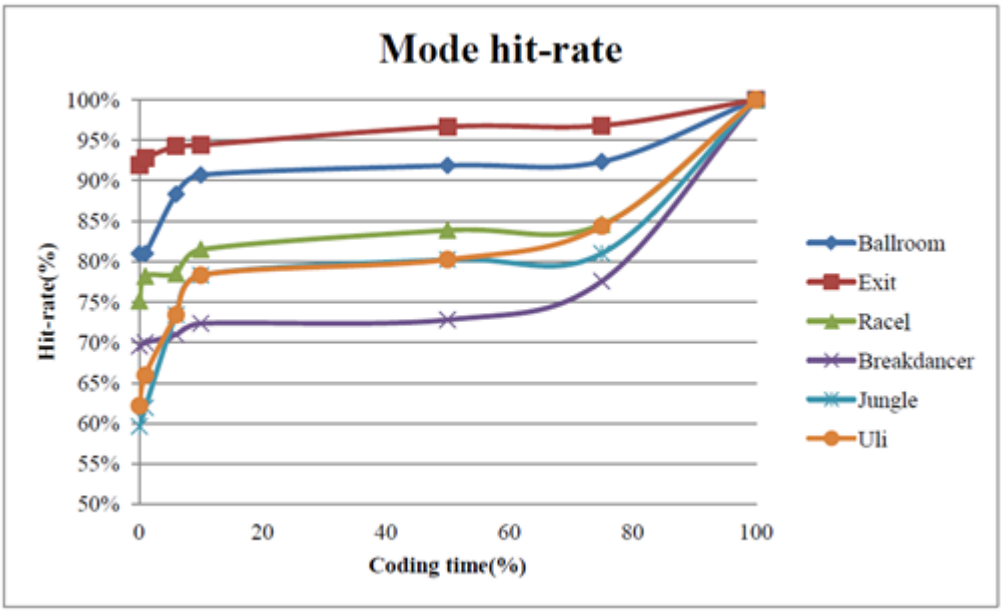

Figure 13: The mode hit-rate

the Mechanism2 receives better performance than Original and Mechanism1. "Original" is the method mentioned before mechanism1 and mechanism2 which MVDs of V2 has the priority to be used.

Table 7: Performance comparisons of the proposed algorithm and CRS plus mode priority for Race1 sequence

QP	PSNR(dB)			Bit-Rate (kbits/s)		
	CRS	CRS+Mode Priority	Proposed	CRS	CRS+Mode Priority	Proposed
28	36.5953	36.6748	36.6862	602.1126	568.7022	552.4267
32	34.3964	34.4752	34.5393	360.3704	332.3644	321.9141
36	32.2517	32.3674	32.3718	242.3970	212.7615	218.5985
40	29.9848	30.0178	30.0261	186.5926	166.6459	167.6563

Table 8: Performance comparisons of the proposed algorithm and CRS plus mode priority

Sequence	BDPSNR(dB)			BDBit-Rate (kbits/s)		
	CRS	CRS+Mode Priority	Proposed	CRS	CRS+Mode Priority	Proposed
Race1	-0.852	-0.277	-0.237	17.74%	4.75%	4.10%

Table 9: Performance comparisons of different PMVD prediction mechanisms

Sequences	Original	Mechanism 1	Mechanism 2
	PSNR(dB)		
Ballroom	34.3891	34.3523	34.4095
Exit	36.8559	36.8009	36.8693
Race1	34.5393	34.5367	34.5639
	Bitrate (kbps)		
Ballroom	319.1926	333.8025	315.1679
Exit	107.9432	111.3556	106.9012
Race1	321.9141	324.1659	319.0963

5 Conclusion

This paper presents a computation allocation algorithm by inter-view motion vector difference calibration for multi-view video coding. First, a global disparity vector normalization algorithm is modified in this paper to improve the accuracy of a global disparity vector so that the modified GDV could be as accurate as possible. After modified GDV, we apply the GDV to predict motion vector difference. In order to predict motion vector difference accurately, we present the better method for predicting motion vector difference. In addition, this paper also proposes a computation allocation algorithm for efficiently dispatching the computation resources for mode decision and motion estimation under computational constraint. Simulation results show that our proposed algorithm can achieve less computational resource requirement with ignorable rate distortion performance degradation and get better performance.

Acknowledgments

This work was supported in part by the Ministry of Science and Technology, Taiwan, R.O.C, under Grant NSC 100-2628-E-259-002-MY3 and MOST 103-2221-E-259-009-MY3.

References

[1] T. Wiegand, G. J. Sullivan, G. Bjontegaard and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.

[2] Y. Chen, P. Pandit and S. Yea, *WD 4 Reference Software for MVC*, ISO/IEC/JTC1/SC29/WG11 and ITU-T Q6/SG16, Doc. JVT-AD207, Jan. 2009.
[3] L. Shen, Z. Liu, P. An, R. Ma, and Z. Zhang, "Low-complexity mode decision for MVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 6, pp. 837–843, June 2011.
[4] Z. Peng, G. Jiang, M. Yu, "A fast multi-view video coding algorithm based on dynamic multi-threshold," in *IEEE International Conference on Multimedia and Expo*, pp. 113–116, 2009.
[5] W. Zhu, X. Tian, F. Zhou and Y. Chen, " Fast disparity estimation using spatio-temporal correlation of disparity field for multiview video coding," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 2, pp. 957–964, 2010.
[6] K. H. Liu, T. J. Liu, and H. H. Liu, "A sift descriptor based method for global disparity vector estimation in multi-view video coding," in *IEEE International Conference on Multimedia and Expo*, pp. 1214–1218, 2010.
[7] L. Shen, Z. Liu, T. Yan, Z. Zhang, and P. An, "View-adaptive motion estimation and disparity estimation for low complexity multiview video coding," *IEEE Transactions on Circuits and Systems for Technology*, vol. 20, no. 6, pp. 925–930, June 2010.
[8] T. Y. Kuo, C. K. Yeh and H. Y. Tsai, "A novel method for global disparity vector estimation in multi-view video coding," in *IEEE International Symposium on Circuits and Systems*, pp. 864–867, 2009.
[9] H. W. Chi, G. L. Li, and M. J. Chen, "Computation-aware algorithm based on inter-view motion vector analysis for multi-view video coding," in *International Computer Science and Engineering Conference*, Bangkok, Thailand, Sept. 2013.
[10] L. W. Zheng, G. L. Li, M. J. Chen, C. H. Yeh, K. H. Tai and J. S. Wu, "Computation controllable mode decision and motion estimation for scalable video coding," *Electronics and Telecommunication Research Institute Journal*, vol. 35, no. 3, pp. 469–479, June 2013.
[11] T. H. Wang, M. J. Chen, M. C. Chi, S. F. Huang and C. H. Yeh, "Computation-scalable algorithm for scalable video coding," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 3, pp. 1194–1202, Aug. 2011.
[12] J. S. Wu, G. L. Li, M. J. Chen, "Effective complexity control by inter-layer motion analysis for spatial scalability video coding," in *IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC'12)*, pp. 78–83, 2012.
[13] X. L. Tang, S. K. Dai and C. H. Cai, "An analysis of TZsearch algorithm in JMVC," in *International Conference on Green Circuits and Systems*, pp. 21–23, Aug. 2010.
[14] G. Bjontegaard, *Calculation of Average PSNR Differences Between RD-curves*, ITU-T SG16 Doc. VCEG-M33, Apr. 2001.

Hao-Wen Chi received the B.S. in Electrical Engineering from Tamkang University, Taipei, Taiwan, in 2011 and received the M.S. degree in Electrical Engineering from National Dong Hwa University, Hualien, Taiwan, in 2013. Currently, he is working for Intellectual Property Office ministry of economic affairs, R.O.C. (TIPO). His research topics include image/video processing, video compression, motion estimation and multi-view video coding.

Gwo-Long Li received his B.S. degree from the Department of Computer Science and Information Engineering, Shu-Te University, Kaohsiung, Taiwan in 2004; M.S. degree from the Department of Electrical Engineering, National Dong Hwa University, Hualien, Taiwan in 2006; and Ph.D. degree from the Department of Electronics Engineering, National Chiao-Tung University, Hsinchu, Taiwan in 2011. During 2011 to 2014, he was an engineer in Industrial Technology Research Institute (ITRI), Hsinchu, Taiwan. In 2006, he received the Excellent Master Thesis Award from Institute of Information and Computer Machinery, Taiwan. He also received the Excellent R&D Substitute Service Award from Ministry of the Interior, Taiwan in 2014. He is currently a senior engineer in Novatek Microelectronics Corp., Hsinchu Taiwan. His research interest is the video signal processing, coding and its VLSI architecture design.

Mei-Juan Chen received her BS, MS, and PhD degrees in Electrical Engineering from National Taiwan University, Taipei, in 1991, 1993, and 1997, respectively. She was an assistant professor (1997-2000) and an associate professor (2000-2005) in the Department of Electrical Engineering, National Dong Hwa University, Hualien, Taiwan. Since August 2005, she has been a professor of the Department of Electrical Engineering, National Dong Hwa University. She also served as the chair of her department from 2005-2006. Her research topics include image/video processing, video compression, motion estimation, error concealment and video transcoding. In 1993 and 1997, she was the recipient of the Dragon Paper Awards and the Xeror Paper Award. She was also the recipient of the 2005 K.T. Li Young Researcher Award from ACM Taipei/Taiwan Chapter for her contribution to video signal codec technique. This award is given annually to only one person under the age of 36, conducting research in Taiwan. In 2006, she received the Distinguished Young Engineer Award from the Chinese Institute of Electrical Engineering, Taiwan. In 2006, she also received the Excellent Master Thesis Supervision Award from the Institute of Information and Computer Machinery. In 2005 and 2012, she received the Jun S. Huang Memorial Foundation Best Paper Awards. She received IPPR society Best Paper Awards in 2013 and 2015. In 2011 and 2014, she received the Best and Excellent Master

Thesis Supervision Awards from Taiwan Institute of Electrical and Electronic Engineering, respectively. In 2014, she received the Best Paper Award from National Symposium on Telecommunication. She has served as associate editor for the EURASIP Journal on Advances in Signal Processing since 2008.

Jie-Ru Lin received his B.S. degree from the Department of Electrical Engineering in National Dong Hwa University, Hualien, Taiwan, in 2015. He is studying M.S. degree in Department of Electrical Engineering in National Dong Hwa University, Hualien, Taiwan. His research interest is video coding algorithm.

Guide for Authors
International Journal of Electronics and Information Engineering

International Journal of Electronics and Information Engineering (IJEIE) will be committed to the timely publication of very high-quality, peer-reviewed, original papers that advance the state-of-the art and applications of Electronics and Information Engineering. Topics will include, but not be limited to, the following: Algorithms, Bioinformatics, Computation Theory, AI, Fuzzy Systems, Embedded Systems, VLSI/EDA, Computer Networks, Information Security, Database, Data Mining, and Information Retrieval, Digital Content, Digital Life, and Human Computer Interaction, Image Processing, Computer Graphics, and Multimedia Technologies, Information Literacy, e-Learning, and Social Media, Mobile Computing, Wireless Communications, and Vehicular Technologies, Parallel, Peer-to-peer, Distributed, and Cloud Computing, Semiconductor, Software Engineering and Programming Languages, Telecommunication, etc.

1. Submission Procedure

Authors are strongly encouraged to submit their papers electronically by using online manuscript submission at http://ijeie.jalaxy.com.tw/.

2. General

Articles must be written in good English. Submission of an article implies that the work described has not been published previously, that it is not under consideration for publication elsewhere. It will not be published elsewhere in the same form, in English or in any other language, without the written consent of the Publisher.

2.1 Length Limitation:

All papers should be concisely written and be no longer than 30 double-spaced pages (12-point font, approximately 26 lines/page) including figures.

2.2 Title page

Title page should contain the article title, author(s) names and affiliations, address, an abstract not exceeding 100 words, and a list of three to five keywords.

2.3 Corresponding author

Clearly indicate who is willing to handle correspondence at all stages of refereeing and publication. Ensure that telephone and fax numbers (with country and area code) are provided in addition to the e-mail address and the complete postal address.

2.4 References

References should be listed alphabetically, in the same way as follows:
For a paper in a journal: M. S. Hwang, C. C. Chang, and K. F. Hwang, ``An ElGamal-like cryptosystem for enciphering large messages," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 2, pp. 445--446, 2002.

For a book: Dorothy E. R. Denning, *Cryptography and Data Security*. Massachusetts: Addison-Wesley, 1982.

For a paper in a proceeding: M. S. Hwang, C. C. Lee, and Y. L. Tang, ``Two simple batch verifying multiple digital signatures," in *The Third International Conference on Information and Communication Security (ICICS2001)*, pp. 13--16, Xian, China, 2001.

In text, references should be indicated by [number].

2.5 Author benefits

No page charge is made.

Subscription Information

Individual subscriptions to IJEIE are available at the annual rate of US$ 200.00 or NT 6,000 (Taiwan). The rate is US$1000.00 or NT 30,000 (Taiwan) for institutional subscriptions. Price includes surface postage, packing and handling charges worldwide. Please make your payment payable to "Jalaxy Technique Co., LTD." For detailed information, please refer to http://ijeie.jalaxy.com.tw or Email to ijeieoffice@gmail.com.