and the unused rakes r-r_1 are carried forward to next call k_2R. Therefore the maximum rakes available to handle second call k_2R are $r_2^{\max} = r + (r\text{-}r_1)$. If this call requires r_2 rakes, the call can be handled without wastage if $r_2 \le r + (r - r_1)$. Similarly, for i^{th} call k_iR, the maximum rakes which can be utilized are

$$r_i^{\max} = i \times r - \sum_{j=1}^{i-1} r_j \quad (6)$$

If the rakes used by i^{th} call are r_i, the rakes carried forward to handle $(i+1)^{\text{th}}$ call are $r_i^{\max} - r_i$. Therefore, in this design the unused rakes of previous calls are utilized by current call. The design always produce zero wastage if r >>1. Even, for nominal value of r (say $L/2$), the algorithm produces zero wastage. The design is particularly useful if the system has large rake consuming rates. The flowchart of the design is shown in Fig. 6, illustrating the procedure to handle i^{th} new call. For illustration of zero wastage designs I, consider a 7 layer code tree shown in Fig. 1. Let the calls arrive in pattern 2R, 16R, 15R, R, R, R, 2R and R. The system is assumed to have 3 rakes. Starting with the first call 2R, the minimum rakes (codes) used is one, and is $C_{2,1}$, as shown in Fig. 7. The identifier $\underline{1}$ around code $C_{2,1}$ signifies handling of 1st call, and the optimum codes $C_{2,1}$ is selected according to CFA [3] design. Similarly, identifier $\underline{x}_y$ around a code represents that the code is handling yR rate fraction of call x. The balance rakes available after handling 1st call are 3-1= 2, and are carried forward to handle the 2nd call. The next call is 16R, which can use 5 (2 pending rakes and 3 regular rakes). Considering minimum rakes usage again, the call is handled by 3 rakes (8+4+4), which are represented by $\underline{2}_8$, $\underline{2}_4$, and $\underline{2}_4$. The relationship in pending rakes, rakes used and rakes carried forward for all calls is given in Table 1. For last two calls 2R and R, the rakes carried forward are 8 and 10, making total rakes available for these calls 11 and 14, which exceeds the maximum rakes required by any call which is L (L is 7 for the example assumed). The status of the code tree after handling all calls is shown in Fig. 7.

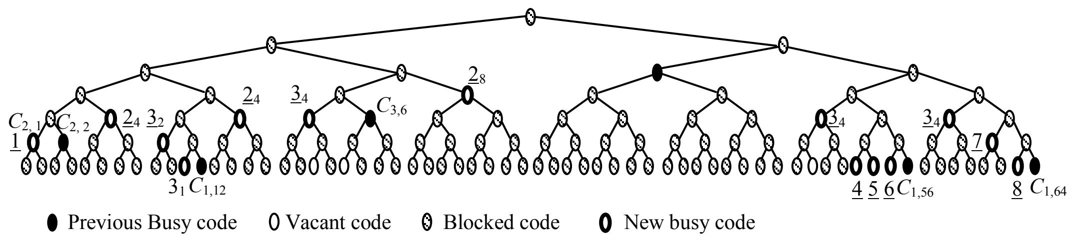


Fig. 7 The updated code tree status for Fig. 1 using design I

Table 1 Rakes usage for examples in Fig. 3 and Fig. 4 in zero wastage and reduced wastage designs

Arrival Rate	Zero wastage design I			Zero wastage design II			Reduced wastage design		
	Pending rakes (say 0 initially) + *r*	Rakes used	Rakes carried forward	Pending rakes (say 0 initially) + *r*	Rakes used	Rakes carried forward	Pending rakes (say 0 initially) + *r*	Rakes used	Rakes carried forward
2*R*	3	1	2	3	1	2	3	1	2
16*R*	5	3	2	5	3	2	5	3	0
15*R*	5	5	0	5	5	0	3	Call rejected	
1*R*	3	1	2	3	1	2	3	1	2
1*R*	5	1	4	5	1	3	5	1	2
1*R*	7	1	6	6	1	3	5	1	2
2*R*	9	1	8	6	1	3	5	1	2
1*R*	11	1	10	6	1	3	5	1	2

3.2 Zero Wastage Design II

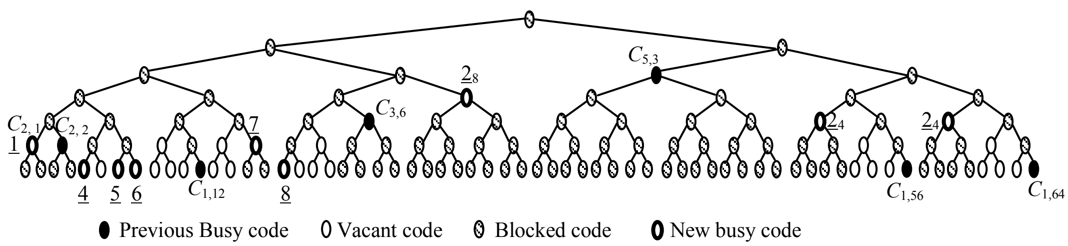


Fig. 8 The updated code tree status for Fig. 1 using reduced wastage design

For an L layer code tree, the maximum rakes required to produce zero wastage are L-1. The design I keep on adding pending rakes even when the total rakes available go beyond L-1. In this design, the number of rakes carried forward can have two possible values: 1) if the pending rakes are $i \times r - \sum_{j=1}^{i-1} r_j \geq L-m-1$, the rakes carried forward to $(i+1)^{th}$ call are L-m-1; 2) if the pending rakes are $i \times r - \sum_{j=1}^{i-1} r_j < L-m-1$, the rakes carried forward to the i^{th} call is $i \times r - \sum_{j=1}^{i-1} r_j$.

Considering the call pattern similar to the one assumed for design I, in zero wastage design II, maximum L-1 (equal to 6) rakes are carried forward to handle next call. The procedure is identical to design till 5^{th} call. The number of rakes carried forward cannot be more than 3 (making pending rakes for new call 6). Hence the pending rakes for all calls after 4^{th} call are different compared to design I as shown in Table 1.

3.3 Reduced Wastage Design

In this design, for a call kR, if minimum rakes required are r_1, the rakes carried forward to the second call are r-r_1. If second call requires minimum r_2 rakes, there are two possible value for amount of rakes carried forward: 1) if $r_2 \leq r$, the rakes carried forward to third call are r-r_2, i.e., all previous pending rakes are discarded; 2) if $r_2 > r$, the maximum rakes used are $r + r_1^{'}, r_1^{'} \leq r - r_1$. The rakes carried forward are $r - r_1 - r_1^{'}$. In general, for i^{th} call, if $r_i \leq r$ arrival the rakes carried forward are r-r_i, otherwise the rakes carried forward are $r - r_i - r_{i-1}^{'}$. The design is simple and cost effective because it requires only previous call rakes information.

For illustration of the reduced wastage design, consider the call arrival pattern similar to the one used

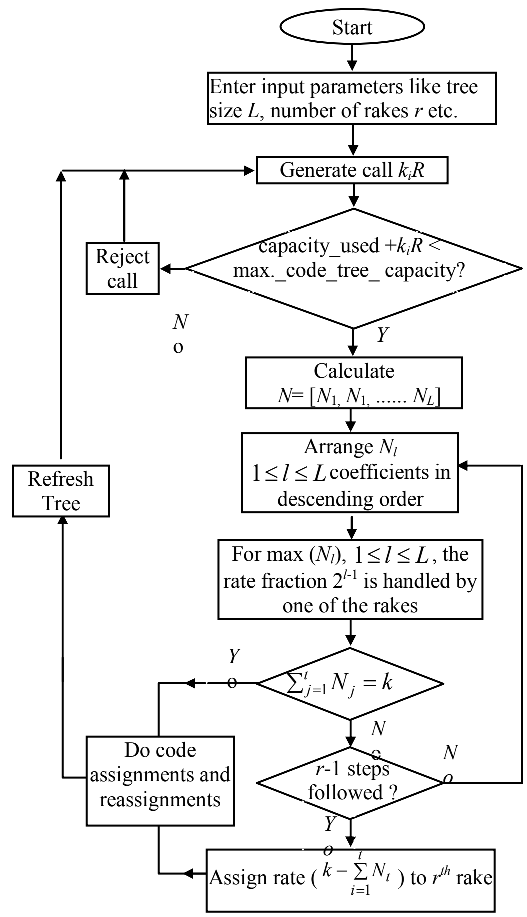


Fig. 9 Flowchart of the proposed multi code scheme

for design I and II, and 3 rakes availability in Fig. 1. The first call with rate $2R$ is handled with one rake as represented by 1 in Fig. 8, and the balance 2 rakes will be carried further to handle the next call. The next call of rate $16R$ has a maximum of 5 rakes as discussed earlier. Using minimum rakes, the call is handled by 3 codes with rate $8R$, $4R$, $4R$ which are $\underline{2}_8$, $\underline{2}_4$ and $\underline{2}_4$. As all the three rakes are consumed for $16R$ call, the number of rakes carried forward is zero. The next call of rate $15R$ requires 4 rakes, and hence is rejected. The next call of rate R now has its own 3 rake quota available. So the call will be handled by 1 rake, which is represented by $\underline{4}$. Similarly, all the remaining calls will be handled by codes represented by $\underline{5}$, $\underline{6}$, $\underline{7}$ and $\underline{8}$. The updated status of the tree is shown in Fig. 8.

3.4 Fair multi code design

The multi code design consists of the three steps: 1) identify the available codes in each layer of the tree; 2) the layers are arranged in descending order of number of vacant codes; 3) use the code which has maximum availability. If the call is not quantized, repeat above two steps until all rate fractions are handled.

For a new call kR, define vacant code vector N giving vacant codes in each layer, i.e., $N = [N_1, N_2, \ldots\ldots N_L]$, where N_l is the number of vacant codes available in l^{th} layer. A vacant code in layer l is included in N_l if all its ancestors are blocked. Arrange coefficients N_l , $1 \le l \le L$ in descending order. Assuming that in the first attempt, the coefficient N_{l_1} is largest, the first rake handles rate fraction $2^{l_1-1}R$ and decrement N_{l_1} by 1. The remaining capacity to be handled by (r-1) rakes is $(2^{l-1} - 2^{l_1-1})R$, and again all the coefficients are arranged in descending order. If N_{l_2} is largest in second attempt, the vacant code in l_2 layer will handle rate fraction $2^{l_2-1}R$. If $2^{l_1-1} + 2^{l_2-1} = k$, procedure stops, otherwise the procedure is repeated to maximum (r-1) times. After (r-1) steps, the fraction of rate kR handled is $\sum_{i=1}^{r-1} 2^{l_i-1}$. If we define, $m = k - \sum_{i=1}^{r-1} 2^{l_i-1}$, find $min(j) | m \le 2^{j-1}$. The rate fraction $2^{j-1}R$ will be handled by r^{th} rake.

The flowchart for the fair multi code design is given in Fig. 9. For illustration of fair multi code design, consider OVSF code tree in Fig. 1 with N=[3,4,5,1,0,0,0], If a new call of $16R$ arrives and the system is equipped with 4 rakes, the combinations which can be used in the mentioned code tree are [$8R$,$4R$,$4R$] and [$4R$,$4R$,$4R$,$4R$] respectively. In both combinations, for each fraction, the algorithm recursively searches for a layer with maximum number of vacant codes. The value of N after utilizing combination I and II is [3,4,3,0,0,0,0] and [3,4,1,1,0,0,0,0] respectively. The combination [3,4,1,1,0,0,0] is preferred as it provides fair distribution of codes for future calls. Using CFA design, the optimum codes used are $C_{4,4}$, $C_{3,2}$ and $C_{3,4}$ respectively as shown in Fig. 10.

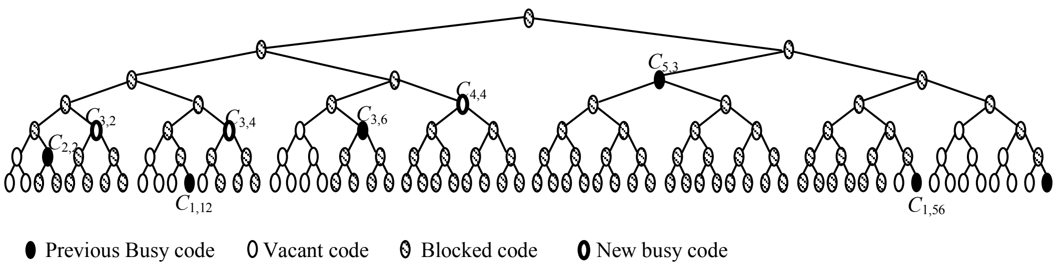


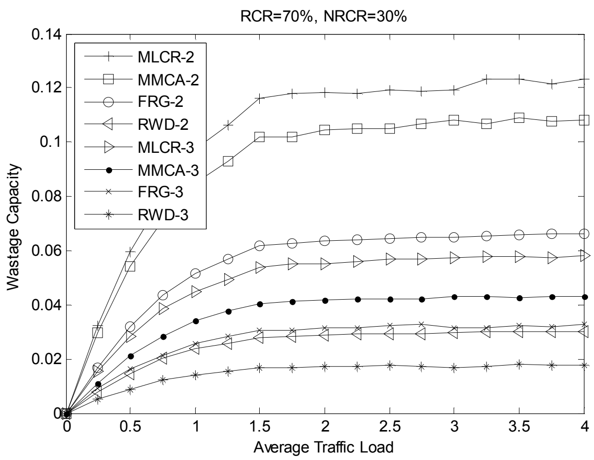
Fig. 10 The updated code tree status for Fig. 1 using fair multi code design

4 Numerical Results

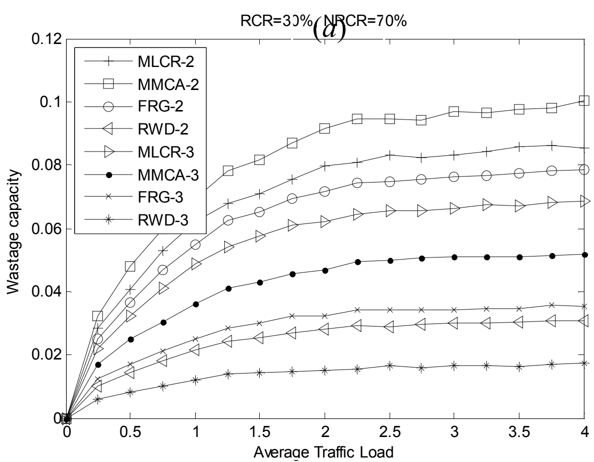
For simulation, five classes of rates namely, R (7.5*kbps*), $2R$, $4R$, $8R$ and $16R$ are considered. The arrival rate for each of the l^{th} class (denoted by λ_l) is assumed to vary between 1 and 4 calls per unit of time, and the call duration ($1/\mu$) is assumed to be 1 unit of time for all classes. If the traffic load for l^{th} class is defined as λ_l/μ, the wastage capacity and other performances are compared for variable average traffic load λ/μ, where, $\lambda = \sum_{l=1}^{5} \lambda_l$. If for a layer l, R_i^l denotes the rate of i^{th} new call (quantized or non-quantized), and M_i^l denotes the sum of capacities of all used codes for this call, the wastage capacity can be rewritten as

$$W_i^l = (M_i^l - R_i^l) / R_i^l \quad (7)$$

where $M_i^l = \sum_{j=1}^{r_i^l} C_{i,j}^l$, and r_i^l is the number of rakes used, and the identifier $C_{i,j}^l$ represents capacity of j^{th} code in the multi code M_i^l. Considering N_l calls for layer l, the total wastage capacity for L layer system is



(*a*)



(*b*)

Fig. 11 Comparison of wastage capacity for distributions, (*a*) *RCR* rates=70%, *NRCR* rates=30%, (*b*) *RCR* rates=30%, *NRCR* rates=70%.

given by

$$W = \sum_{l=1}^{L} \sum_{i=1}^{N_l} W_i^l = \sum_{l=1}^{L} \sum_{i=1}^{N_l} (M_i^l - R_i^l) / R_i^l \quad (8)$$

The first performance parameter considered is the code wastage capacity, and the comparison for the proposed reduced wastage design (represented by RWG) is done with multicode fragmentation (FRG) [1], multiple leaf code reservation (MLCR) [10], and multicode multirate compact assignment (MMCA) [12] designs discussed in section 1. The results are plotted in Fig. 11(*a*) and 11(*b*), in terms of rake consuming and non rake consuming rates only. Two distributions of *RCR* and *NRCR* rates are considered: 1) the probability of *RCR* and *NRCR* rates arrival is

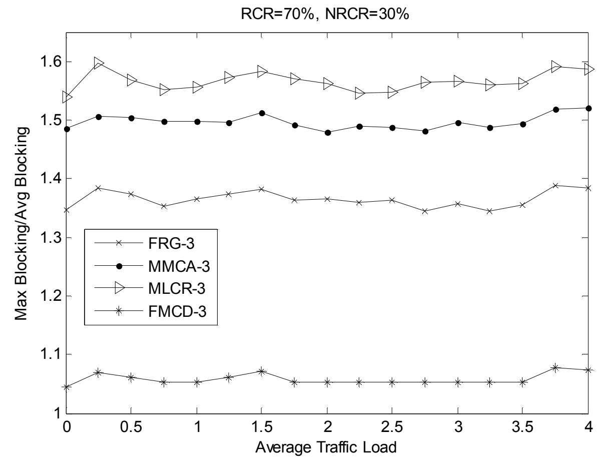


Fig. 12 Comparison of ratio of maximum blocking to average blocking for distribution, *RCR* rates=70%, *NRCR* rates=30%.

70% and 30%; 2) the probability of *RCR* and *NRCR* rates arrival is 30% and 70%. In Fig. 11, the symbol *des-n*, represents *n* rakes availability in design *des*, e.g., FRG-3, represents the fragmentation design with 3 rakes availability. The wastage capacity results Fig. 11(*a*) and 11(*b*) shows that the wastage capacity in the proposed design is significantly less than the existing alternatives. Further, comparing Fig. 11(*a*) and 11(*b*), the wastage capacity in the system dominated by *RCR* rates scenario is slightly more than the one with dominating *NRCR* rates scenario. Also, the wastage capacity reduces as the number of rakes in the system increases. The zero wastage design results are not plotted because they produce zero wastage.

The fairness in handling various rate users is also plotted in Fig. 12 for proposed fair multi code design (FMCD) with above mentioned schemes when the system has three rakes. The parameter maximum_blocking/average_blocking is used for fairness comparisons, and for perfect fair system, this parameter should have unit value. The results are plotted for the rate distribution *RCR*=70%, and

NRCR=30%, which clearly shows that the proposed FMCD design is, by far, the most fair design.

5. Conclusion

The use of multiple codes in OVSF based CDMA always gives better results in handling non-quantized rates. Traditional multi code designs use either minimum rakes or maximum rakes for new rates. The use of more codes increases cost and complexity. The proposed multi code design uses the balanced rakes of quantized or closely quantized calls to handle non-quantized calls designated as rake consuming rates in the paper. Only few extra channels need to be carried forward to new calls. The wastage capacity is drastically reduced giving reduction in number of calls rejected. The increase in complexity in this zero blocking design can be reduced by suboptimal design with little compromise in increase of wastage capacity. Work can be done to make rakes carried forward adaptive to the call arrival distribution.

References:

[1] C. M. Chao, Y. C. Tseng and L. C. Wang, Reducing Internal and External Fragmentation of OVSF Codes in WCDMA Systems with Multiple Codes, *IEEE Transactions on Wireless Communications*, Vol. 4, Jul. 2005, pp. 1516-1526,

[2] F. Adachi, M. Sawahashi, and K. Okawa, Tree-structured generation of orthogonal spreading codes with different lengths for forward link of DS-CDMA mobile radio, *Electronic Letters*, Vol. 33, No. 1, Jan. 1997, pp. 27–28.

[3] Y. C. Tseng and C. M. Chao, Code placement and replacement strategies for wideband CDMA OVSF code tree management, *IEEE Transactions on Mobile Computing,* Vol. 1, No. 4, 2002, pp. 293–302.

[4] J. S. Park and D. C. Lee, Enhanced fixed and dynamic code assignment policies for OVSF-CDMA systems, *Proc. ICWN*, Jun. 2003, pp. 620–625.

[5] T. Minn and K. Y. Siu, Dynamic assignment of orthogonal variable spreading factor codes in W-CDMA, *IEEE Journal on Selected Areas in Communications*, Vol. 18, No. 8, Aug. 1998, pp. 1429–1440.

[6] J. S. Park, L. Huang, and C. C. J. Kuo, Computationally efficient dynamic code assignment schemes with call admission control (DCA-CAC) for OVSF-CDMA systems, *IEEE Transactions on Vehicular Technology*, Vol. 54, No. 1, Jan. 2008, pp. 286–296.

[7] A. Rouskas and D. Skoutas, OVSF code assignment and reassignment at the forward link of W-CDMA 3G systems, in *Proc. IEEE PIMRC*, Sept.2002, pp.2404-2408.

[8] A. N . Rouskas and D. N. Skoutas, Management of channelization codes at the forward link of WCDMA, *IEEE Commun. Lett.*, vol. 9, Aug.2005, pp. 679-681.

[9] V. Balyan and D. S. Saini, Call Elapsed Time and reduction in code blocking for WCDMA Networks, in *Proc. Softcom* 2009, Croatia, Sept. 2009, pp. 1-5.

[10] F. A. Cruz-Perez, J. L. Vazquez-Avila, A. Seguin-Jimenez, and L. Ortigoza-Guerrero, Call Admission and Code Allocation Strategies for WCDMA Systems With Multirate Traffic, *IEEE Journal on Selected Areas on Communications*, Vol.24, Jan. 2006, pp. 26-35.

[11] D. S. Saini and M. Upadhyay, Multiple rake combiners and performance improvement in WCDMA systems, *IEEE Transactions on Vehicular Technology*, Vol. 58, No. 7, Sep. 2009, pp. 3361-3370.

[12] Y. Yang and T. S. P. Yum, Multicode Multirate Compact Assignment of OVSF Codes for QoS Differentiated Terminals, *IEEE Transactions on Vehicular Technology*, Vol. 54, No. 6, Nov. 2005, pp. 2114-2124.

[13] M. X. Chen, Efficient integration OVSF code management architecture in UMTS, *Elsevier Computer Communications*, Vol. 31, No. 9, pp. 3103–3112, Sep. 2008.

[14] S. T. Cheng and M. T. Hsieh, Design and analysis of time-based code allocation schemes in W-CDMA systems, *IEEE Transactions on Mobile Computing*, Vol. 4, No. 6, Nov./Dec. 2005, pp. 604-615.

[15] C. M. Chao, OVSF Code Assignment Strategies with Minimal Fragmentations for WCDMA Systems, *Computer Networks,* Vol. 52, No. 12, Aug. 2008, pp. 2331-2343.

Appendix A

Table A.1 Relationship between code wastage capacity and rakes for various user rates in WCDMA

Rakes / Rate (Rbps)	N=1	2	3	4	5	6	7		Rakes / Rate (Rbps)	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0		65	63/128	0	0	0	0	0	0
2	0	0	0	0	0	0	0		66	62/128	0	0	0	0	0	0
3	1/4	0	0	0	0	0	0		67	61/128	1/68	0	0	0	0	0
4	0	0	0	0	0	0	0		68	60/128	0	0	0	0	0	0
5	3/8	0	0	0	0	0	0		69	59/128	3/72	0	0	0	0	0
6	2/8	0	0	0	0	0	0		70	58/128	2/72	0	0	0	0	0
7	1/8	1/8	0	0	0	0	0		71	57/128	1/72	1/72	0	0	0	0
8	0	0	0	0	0	0	0		72	56/128	0	0	0	0	0	0
9	7/16	0	0	0	0	0	0		73	55/128	7/80	0	0	0	0	0
10	6/16	0	0	0	0	0	0		74	54/128	6/80	0	0	0	0	0
11	5/16	1/12	0	0	0	0	0		75	53/128	5/80	1/76	0	0	0	0
12	4/16	0	0	0	0	0	0		76	52/128	4/80	0	0	0	0	0
13	3/16	3/16	0	0	0	0	0		77	51/128	3/80	3/80	0	0	0	0
14	2/16	2/16	0	0	0	0	0		78	50/128	2/80	2/80	0	0	0	0
15	1/16	1/16	1/16	0	0	0	0		79	49/128	1/80	1/80	1/80	0	0	0
16	0	0	0	0	0	0	0		80	48/128	0	0	0	0	0	0
17	15/32	0	0	0	0	0	0		81	47/128	15/96	0	0	0	0	0
18	14/32	0	0	0	0	0	0		82	46/128	14/96	0	0	0	0	0
19	13/32	1/20	0	0	0	0	0		83	45/128	13/96	1/84	0	0	0	0
20	12/32	0	0	0	0	0	0		84	44/128	12/96	0	0	0	0	0
21	11/32	3/24	0	0	0	0	0		85	43/128	11/96	3/88	0	0	0	0
22	10/32	2/24	0	0	0	0	0		86	42/128	10/96	2/88	0	0	0	0
23	9/32	1/24	1/24	0	0	0	0		87	41/128	9/96	1/88	1/88	0	0	0
24	8/32	0	0	0	0	0	0		88	40/128	8/96	0	0	0	0	0
25	7/32	7/32	0	0	0	0	0		89	39/128	7/96	7/96	0	0	0	0
26	6/32	6/32	0	0	0	0	0		90	38/128	6/96	6/96	0	0	0	0
27	5/32	5/32	1/28	0	0	0	0		91	37/128	5/96	5/96	1/92	0	0	0
28	4/32	4/32	0	0	0	0	0		92	36/128	4/96	4/96	0	0	0	0
29	3/32	3/32	3/32	0	0	0	0		93	35/128	3/96	3/96	3/96	0	0	0
30	2/32	2/32	2/32	0	0	0	0		94	34/128	2/96	2/96	2/96	0	0	0
31	1/32	1/32	1/32	1/32	0	0	0		95	33/128	1/96	1/96	1/96	1/96	0	0
32	0	0	0	0	0	0	0		96	32/128	0	0	0	0	0	0
33	31/64	0	0	0	0	0	0		97	31/128	31/128	0	0	00	0	0
34	30/64	0	0	0	0	0	0		98	30/128	30/128	0	0	0	0	0
35	29/64	1/36	0	0	0	0	0		99	29/128	29/128	1/100	0	0	0	0
36	28/64	0	0	0	0	0	0		100	28/128	28/128	0	0	0	0	0
37	27/64	3/40	0	0	0	0	0		101	27/128	27/128	3/104	0	0	0	0
38	26/64	2/40	0	0	0	0	0		102	26/128	26/128	2/104	0	0	0	0
39	25/64	1/40	1/40	0	0	0	0		103	25/128	25/128	1/104	1/104	0	0	0
40	24/64	0	0	0	0	0	0		104	24/128	24/128	0	0	0	0	0
41	23/64	7/48	0	0	0	0	0		105	23/128	23/128	7/112	0	0	0	0
42	22/64	6/48	0	0	0	0	0		106	22/128	22/128	6/112	0	0	0	0
43	21/64	5/48	1/44	0	0	0	0		107	21/128	21/128	5/112	1/108	0	0	0
44	20/64	4/48	0	0	0	0	0		108	20/128	20/128	4/112	0	0	0	0
45	19/64	3/48	3/48	0	0	0	0		109	19/128	19/128	3/112	3/112	0	0	0
46	18/64	2/48	2/48	0	0	0	0		110	18/128	18/128	2/112	2/112	0	0	0
47	17/64	1/48	1/48	1/48	0	0	0		111	17/128	17/128	1/112	1/112	1/112	0	0
48	16/64	0	0	0	0	0	0		112	16/128	16/128	0	0	0	0	0
49	15/64	15/64	0	0	0	0	0		113	15/128	15/128	15/128	0	0	0	0
50	14/64	14/64	0	0	0	0	0		114	14/128	14/128	14/128	0	0	0	0
51	13/64	13/64	1/52	0	0	0	0		115	13/128	13/128	13/128	1/116	0	0	0
52	12/64	12/64	0	0	0	0	0		116	12/128	12/128	12/128	0	0	0	0
53	11/64	11/64	3/56	0	0	0	0		117	11/128	11/128	11/128	3/120	0	0	0
54	10/64	10/64	2/56	0	0	0	0		118	10/128	10/128	10/128	2/120	0	0	0
55	9/64	9/64	1/56	1/56	0	0	0		119	9/128	9/128	9/128	1/120	1/120	0	0
56	8/64	8/64	0	0	0	0	0		120	8/128	8/128	8/128	0	0	0	0
57	7/64	7/64	7/64	0	0	0	0		121	7/128	7/128	7/128	7/128	0	0	0
58	6/64	6/64	6/64	0	0	0	0		122	6/128	6/128	6/128	6/128	0	0	0
59	5/64	5/64	5/64	1/60	0	0	0		123	5/128	5/128	5/128	5/128	1/124	0	0
60	4/64	4/64	4/64	0	0	0	0		124	4/128	4/128	4/128	4/128	0	0	0
61	3/64	3/64	3/64	3/64	0	0	0		125	3/128	3/128	3/128	3/128	3/128	0	0
62	2/64	2/64	2/64	2/64	0	0	0		126	2/128	2/128	2/128	2/128	2/128	0	0
63	1/64	1/64	1/64	1/64	1/64	0	0		127	1/128	1/128	1/128	1/128	1/128	1/128	0
64	0	0	0	0	0	0	0		128	0	0	0	0	0	0	0

Davinder S Saini was born in Nalagarh, India in January 1976. He received B.E degree in electronics and telecommunication engineering from College of Engineering Osmanabad, India in 1998. He received M.Tech degree in communication systems from Indian Institute of Technology (IIT) Roorkee, India in 2001. He received PhD degree in electronics and communication from Jaypee University of Information Technology Waknaghat, India in 2008.

He is with Jaypee University of Information Technology Waknaghat since June 2002, and he is currently working as an Associate Professor in electronics and communication department. His research areas include Channelization (OVSF) codes and optimization in WCDMA, routing algorithms and security issues in MANETs.