27: OVSF

Friday, February 25, 2011
10:08 AM

3G (IMT-2000) → UMTS (W-CDMA), cdma2000 } both are based on CDMA

OVSF codes
- O → orthogonal ↑ use orthogonal codes
- V → variable ← the code will be of variable length
- SF → spreading factor ← variable processing gain

= variable-length Walsh codes.

Recall CDMA

Sync. CDMA

Hadamard Matrix

$$\begin{matrix} 1 & 1 \\ 1 & -1 \end{matrix} \rightarrow \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{matrix} \begin{matrix} \underline{c}_1 \\ \underline{c}_2 \\ \underline{c}_3 \\ \underline{c}_4 \end{matrix}$$

spread codes

Walsh codes (orderded / arranged by # transitions)

$$\begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{matrix}$$

(OVSF ↑ will revisit this after CDMA review

$$\begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{matrix}$$
)

← observe that the rows are the same as the two matrices above but in different order.

Back to CDMA ...

Suppose you have 2 users

user 1 uses code $\underline{c}_1$

wants to send message $a_1, a_2, a_3, a_4, \ldots$

send $\underline{x}_1 = [a_1 \times \underline{c}_1 \quad a_2 \times \underline{c}_1 \quad a_3 \times \underline{c}_1 \quad a_4 \times \underline{c}_1 \quad \ldots]$

user 2 uses code $\underline{c}_2$

wants to send message $b_1, b_2, b_3, b_4, \ldots$

send $\underline{x}_2 = [b_1 \times \underline{c}_2 \quad b_2 \times \underline{c}_2 \quad b_3 \times \underline{c}_2 \quad \cdots \quad]$

Receiver gets $\underline{r} = \underline{x}_1 + \underline{x}_2$

to recover a_1, calculate $\frac{1}{4}\langle \underline{r}(1:4), \underline{c}_1 \rangle$

assuming there is no noise.

b_1, $\frac{1}{4}\langle \underline{r}(1:4), \underline{c}_2 \rangle$

a_2, $\frac{1}{4}\langle \underline{r}(5:8), \underline{c}_1 \rangle$

For all these to work, need $\underline{c}_1 \perp \underline{c}_2$!

$\langle \underline{c}_1, \underline{c}_2 \rangle = 0$

In the above example, $N = 4$ = processing gain = spreading factor

length of the code

Remark: , all codes have the same length.

Q: Can we have variable-length codes?? and use them simultaneously?

A: Yes.

OVSF-based system

See slides on using tree to build OVSF codes.

Suppose...

User 1 uses $\underline{c}_{4,2}$ → $[1 \ -1 \ 1 \ -1]$

wants to send $a_1, a_2, a_3, \ldots$

send $\underline{x}_1 = [[a_1 \times \underline{c}_{4,2} \ ] \ [a_2 \times \underline{c}_{4,2} \ ] \ [a_3 \times \underline{c}_{4,2}] \cdots$

User 2 uses $\underline{c}_{8,7}$

$[1 \ -1 \ -1 \ 1 \ -1 \ 1 \ 1 \ -1]$

wants to send $b_1, b_2, b_3, \ldots$

send $\underline{x}_2 = [[\quad b_1 \times \underline{c}_{8,7} \quad][\quad b_2 \times \underline{c}_{8,7} \ \cdots$

Remark: If the "chip" duration are the same, then the longer the code (SF), the slower the data rate.

At receiver,

$\underline{r} = \underline{x}_1 + \underline{x}_2$

to recover a_1, compute $\frac{1}{4}\langle \underbrace{\underline{r}(1:4)}, \underline{c}_{4,2}\rangle$

assuming there is no noise

$\underline{x}_1(1:4) + \underline{x}_2(1:4)$

$= a_1 \times \underline{c}_{4,2} + b_1 \underline{c}_{8,7}(1:4)$

Observe that $\langle a_1\underline{c}_{4,2} + b_1 \underline{c}_{8,7}(1:4), \underline{c}_{4,2}\rangle$

$= a_1\langle \underline{c}_{4,2}, \underline{c}_{4,2}\rangle + b_1\langle \underline{c}_{8,7}(1:4), \underline{c}_{4,2}\rangle.$

Hence, to recover a_1, need $\langle \underline{c}_{8,7}(1:4), \underline{c}_{4,2}\rangle = 0$

$\Rightarrow$ need $\underline{c}_{4,2} \perp \underline{c}_{8,7}(1:4)$ (1)

to recover b_1, compute $\frac{1}{8}\langle \underline{r}(1:8), \underline{c}_{8,7}\rangle$

$\underline{x}_1(1:8) + \underline{x}_2(1:8)$

$= [a_1\underline{c}_{4,2} \quad a_2\underline{c}_{4,2}] + b_1\underline{c}_{8,7}$

To get a better idea of the inner product above, recall that, for real-valued row vectors,

$\langle \underline{a}, \underline{b}\rangle = \underline{a}\,\underline{b}^T$

Now, if $\underline{a} = [\,\boxed{\underline{x}_1} \;\; \boxed{\underline{x}_2}\,]$

and

$\underline{b} = [\,\boxed{\underline{y}_1} \;\; \boxed{\underline{y}_2}\,]$

then

$\langle \underline{a}, \underline{b}\rangle = \langle [\,\boxed{\underline{x}_1} \;\; \boxed{\underline{x}_2}\,], [\,\boxed{\underline{y}_1} \;\; \boxed{\underline{y}_2}\,]\rangle$

$= [\,\underline{x}_1 \;\; \underline{x}_2\,]\begin{bmatrix}\underline{y}_1^T \\ \underline{y}_2^T\end{bmatrix}$

$= \underline{x}_1\underline{y}_1^T + \underline{x}_2\underline{y}_2^T = \langle \underline{x}_1, \underline{y}_1\rangle + \langle \underline{x}_2, \underline{y}_2\rangle$

block matrix multiplication.

Hence,

$\langle \underline{r}(1:8), \underline{c}_{8,7}\rangle$

$= \langle [a_1\underline{c}_{4,2} \quad a_2\underline{c}_{4,2}] + b_1\underline{c}_{8,7}, \underline{c}_{8,7}\rangle$

$= \langle [\quad\quad], \underline{c}_{8,7}\rangle + b_1\langle \underline{c}_{8,7}, \underline{c}_{8,7}\rangle$

$$= \langle [\quad], \underbrace{\underline{c}_{8,7}}_{=\,[\underline{c}_{8,7}(1:4)\quad \underline{c}_{8,7}(5:8)]} \rangle + b_1 \langle \underline{c}_{8,7}, \underline{c}_{8,7} \rangle$$

$$= a_1 \langle \underline{c}_{4,2}, \underline{c}_{8,7}^{(1:4)} \rangle + a_2 \langle \underline{c}_{4,2}, \underline{c}_{8,7}(5:8) \rangle$$

$$+ b_1 \langle \underline{c}_{8,7}, \underline{c}_{8,7} \rangle$$

Therefore, to recover b_1, we need $\underline{c}_{4,2} \perp \underline{c}_{8,7}(1:4)$ 1

$\underline{c}_{4,2} \perp \underline{c}_{8,7}(5:8)$ 2

Similarly, you may check that

to recover a_2, we need $\underline{c}_{4,2} \perp c_{8,7}(5:8)$ 2

to recover a_3, we need $\underline{c}_{4,2} \perp c_{8,7}(1:4)$ 1

$\vdots$

and that

to recover b_2, we need $\underline{c}_{4,2} \perp c_{8,7}(1:4)$ 1

$\underline{c}_{4,2} \perp c_{8,7}(5:8)$ 2

$\vdots$

Hence, the conditions for $\underline{c}_{4,2}$ and $\underline{c}_{8,7}$ that are needed for user 1 and user 2 to transmit simultaneously are

$$\left.\begin{array}{l} \underline{c}_{4,2} \perp c_{8,7}(1:4) \\ \underline{c}_{4,2} \perp c_{8,7}(5:8) \end{array}\right\} \text{This will be our definition of } \underline{c}_{4,2} \perp \underline{c}_{8,7}$$

When codes are defined using the tree structure as provided in the slides, we have an easy way to tell whether $\underline{c}_{4,2} \perp \underline{c}_{8,7}$.