Fast convergence and reduced complexity receiver design for LDS-OFDM system

Lei Wen, Razieh Razavi, Pei Xiao and Muhammad Ali Imran
Centre for Communication Systems Research
University of Surrey
Guildford GU2 7XH, Surrey, United Kingdom
Email: {l.wen, r.razavi, p.xiao, m.imran}@surrey.ac.uk

Abstract—Low density signature for OFDM (LDS-OFDM) is able to achieve satisfactory performance in overloaded conditions, but the existing LDS-OFDM has the drawback of slow convergence rate for multiuser detection (MUD) and high receiver complexity. To tackle these problems, we propose a serial schedule for the iterative MUD. By doing so, the convergence rate of MUD is accelerated and the detection iterations can be decreased. Furthermore, in order to exploit the similar sparse structure of LDS-OFDM and LDPC code, we utilize LDPC codes for LDS-OFDM system. Simulations show that compared with existing LDS-OFDM, the LDPC code improves the system performance.

Keywords—LDS-OFDM; iterative multiuser detection; serial schedule; forward error correction

I. INTRODUCTION

Multi-carrier code division multiple access (MC-CDMA) is a multiple access method used in OFDM-based systems, allowing the system to support multiple users at the same time [1]. It has been considered as a suitable approach to coping with increasing data rate in wireless communications [2]. However, when the number of users or parallel data symbols exceeds that of available chips, which is referred to as the overloaded condition, the performance of MC-CDMA degrades dramatically. In that condition, multipath fading becomes a severe problem and the orthogonality of spreading sequences is destroyed, thus the system performance is limited by serious multiuser interference (MUI) and inter-symbol interference (ISI). Classic multiuser detection (MUD) fails to eliminate the MUI in overloaded conditions [3][4]. In order to deal with the problem, low density signature (LDS) for MC-CDMA, namely LDS-OFDM, has been proposed [5][6][7].

In LDS-OFDM, due to the low density signature, each data symbol is only spread over a limited number of chips (effective processing gain), and each chip is transmitted over an orthogonal sub-carrier. Each sub-carrier is only used by a limited number of data symbols that may belong to different users. Consequently, each user, transmitting on given sub-carriers, will experience interference from only a small number of other users’ data symbols. By applying message passing algorithm, the LDS-OFDM shows satisfactory performance and outperforms similar well-known systems over multi-path fading channels. But the receiver complexity of LDS-OFDM is relatively high. Therefore, it is challenging to design a LDS-OFDM receiver to achieve satisfactory performance while reducing complexity. Message passing schedule on sparse graphs not only influences the convergence rate, but also affects the system performance. In [7] and [8], floating schedule is adopted for MUD of LDS-OFDM, where all the nodes update messages simultaneously. Its convergence behavior is not ideal. In this paper we develop a serial schedule to perform MUD. By using more reliable information, the serial schedule improves the convergence rate and reduces receiver complexity. In addition, inspired by sparse structure of LDS-OFDM, we utilize LDPC codes for forward error correction (FEC) in LDS-OFDM system.

The structure of this paper is organized as follows. Section II introduces the system model of LDS-OFDM. In section III, flooding schedule and serial schedule for iterative MUD are presented. LDPC decoding algorithm adopted by LDS-OFDM receiver is presented in section IV. Section V shows simulation results and comparisons. Finally, conclusions are drawn in section VI.

II. SYSTEM MODEL

In this section, a single cell uplink LDS-OFDM system model is presented. Let $k$ be the number of users, $m$ be the identical number of modulated symbols transmitted from each user, and $N$ be the number of chips. Each chip is transmitted over an orthogonal sub-carrier, the ISI can be avoided by the insertion of cyclic prefix (CP) provided that the length of CP is longer than the channel delay spread.

Fig. 1 shows transmitter block diagram of an uplink LDS-OFDM system. Each user has an independent link as shown in the figure. Denote $x_{k,m}$ as the $m$th data symbol of user $k$ ($k \in [1, K], m \in [1, M]$), $c_n$ is the $n$th chip ($n \in [1, N]$). It can be seen that similar to the MC-CDMA spreading process, after FEC encoding and symbol mapping, we multiply the modulated symbol with a spreading signature (a random sequence of chips) and perform the OFDM modulation afterwards. However, in the LDS-OFDM case, the main difference is that the spreading signature has a low density (a large number of chips in the sequence are equated to zero). In other words, the number of users’ modulated symbols that superimposed on each chip is much less than the total number of modulated symbols, $d_m < (K \times M)$, where $d_m$ is the number of symbols that superimposed on one chip. Similarly, the number of chips that spread by each symbol is much less than the total number of chips, $d_n < N$, where $d_n$ is the number of chips that spread by one symbol. In fact, $d_m$ is the effective spreading factor. If $d_m$ and $d_n$ are both constant values, we refer it as a regular LDS-OFDM, otherwise it is an irregular LDS-OFDM.
Fig. 1. Transmitter structure of LDS-OFDM

Fig. 2 shows receiver block diagram of an uplink LDS-OFDM. The receiver is a single base station in practical systems. Users’ signals that are using the same chip will be superimposed. As the number of symbols that interfere with each other at one chip is much less than total number of symbols, the LDS-OFDM can perform well under overload conditions. In Fig. 2, we can see that the dispreading is performed over a low density signature (the dashed rectangle). There are two types of nodes in the signature: chip node $c_i$ and variable node $x_{i,u}$. Different types of nodes are connected by sparse edges. Message passing algorithm can be employed for MUD of LDS-OFDM, which is presented in the next section.

III. ITERATIVE MULTIUSER DETECTION

The spreading matrix for user $k$ is $S_k = [s_{k,1},...,s_{k,M}] \in \mathbb{C}^{N_S \times M}$, where $C$ represents the complex field and $S_k$ has only $d_k$ non-zero elements at each column. Let us denote $S = [s_1,...,s_k] \in \mathbb{C}^{N_S \times K}$ as the low density signature matrix of the LDS-OFDM system, $A = \text{diag}(A_1,...,A_K)$ as the transmit power gain of users and $G_i = \text{diag}(g_{k,1},...,g_{k,M})$ as the corresponding channel gain for the $k^{th}$ user.

In LDS-OFDM, each user’s generated chip will be transmitted over an orthogonal sub-carrier. The received spreading sequence for data symbol $m$ of user $k$ can be represented by $r_{i,u} = T_{G_k} s_{k,u}$. To be more specific, the received signature gain at chip $n$ of data symbol $m$ of user $k$ is $r_{i,u} = T_{G_k} s_{k,u} s_{k,u}$. Let $\psi_k = \{ (k,m) : s_{k,u} \neq 0 \}$ be the set of different symbols (which may belong to different users) that interfere on chip $c_i$, and $\epsilon_k = \{ n : s_{k,u} \neq 0 \}$ be the set of different chips that the $m^{th}$ symbol of user $k$ is spread on.

For an uplink MC-CDMA system, the received signal at chip (sub-carrier) index $n$ is written as

$$y_n = \sum_{k=1}^{K} \sum_{m \in \psi_k} r_{i,u}^m s_{k,u} + z_n$$

where $z_n$ is the noise over chip $c_i$. Because the spreading signature only has a limited number of non-zero positions in LDS-OFDM, we can express the received signal at the $n^{th}$ chip (sub-carrier) as

$$y_n = \sum_{(k,m) \in \epsilon_k} r_{i,u}^m s_{k,u} + z_n$$

Let $L_{i,u}^{j} = \log (\text{likelihood ratio (LLR)}$ delivered from variable node $v_{i,u}$ to chip node $c_i$ at $j^{th}$ iteration. Similarly, the LLR delivered from chip node $c_i$ to variable node $v_{i,u}$ at $j^{th}$ iteration is given by $L_{i,u}^{j-1}$, and $L_{i,u}^{j}$ is the final estimation of variable node $v_{i,u}$. In message passing algorithm, the schedule is a key that determines convergence rate. The flooding schedule has been applied for LDS-OFDM in [7] and [8], which is summarized as follows.

A. Flooding Schedule

- Initialization
  Assuming there is no a priori probability available, initial values at the first iteration are set to zeros:

$$L_{i,u}^{1} = 0 \forall k, \forall m, \forall n$$

- Chip nodes updating
  $L_{i,u}^{j} = f(x_{i,u} | y_n, L_{i,u}^{j-1}, (k,m) \in \psi_k \setminus (k,m))$

where $\psi_k \setminus (k,m)$ is the set of data symbols (excluding $x_{i,u}$) that interfere on chip $c_i$.

In order to approximate the optimum maximum a posteriori probability (MAP) detector, the right hand side of (4) represents marginalization function, which is based on (2), and can be written as

$$f(x_{i,u} | y_n, L_{i,u}^{j-1}, (k,m) \in \psi_k \setminus (k,m)) = \log (\sum_{x_{i,u}} p_x^{j-1}(x_{i,u}) p_{y|x}^{j-1}(x_{i,u}))$$

where the conditional probability density function (PDF) $p_x^{j-1}(x_{i,u})$ and a priori probability $p_{y|x}^{j-1}(x_{i,u})$ are given as

$$p_x^{j-1}(x_{i,u}) = \frac{1}{2\sigma^2} \exp (-\frac{1}{2\sigma^2} \| y_n - r_{i,u}^T v_{i,u} \|^2)$$

and

$$p_{y|x}^{j-1}(x_{i,u}) = \exp (L_{i,u}^{j-1}(x_{i,u}))$$

where $v_{i,u}$ and $r_{i,u}$ denote the vector containing the symbols transmitted by the users that spread their data on chip $n$ and
their corresponding effective received signature values, respectively. As can be seen in (5) that based on received chip $y_v$ and a priori input information $p(v_m|x_v)$, extrinsic values are calculated for all the constituent bits involved in (2). Combining (6) and (7) into (5), the message update will be

$$L_{i,v}^{j} \rightarrow r_{i,v} = \kappa_{i,v} \max ( \sum_{(i') \neq i} L_{i',v}^{j-1} - \frac{1}{2\sigma^2} || y_v - r_i^{v} y_{i'}^{v} ||^2 )$$

(8)

where $\kappa_{i,v}$ denotes the normalization coefficient and

$$\max (a,b) = \log(e^a + e^b)$$

(9)

- Variable nodes updating

$$L_{i,v}^{j} \rightarrow r_{i,v} = \sum_{e_{i,v}^{j-1} \in \varepsilon} L_{e_{i,v}^{j-1} \rightarrow r_{i,v}}$$

(10)

where $\varepsilon_{i,v} \setminus n$ is the set of different chips(excluding $c_v$) that the $m^a$ symbol of user $k$ is spread on.

- Estimation

This technique is based on log-MAP detection. After the message-passing has converged or has reached the maximum number of iterations $J$, a posteriori probability of the transmitted symbol $v_{i,v}$ is estimated as

$$L_{i,v} = \sum_{e_{i,v}^{j-1} \in \varepsilon} L_{e_{i,v}^{j-1} \rightarrow r_{i,v}}$$

(11)

By making a hard decision, the estimated value of $v_{i,v}$ is

$$v_{i,v} = \arg \max_{v_{i,v}} L_{i,v} (v_{i,v})$$

(12)

### B. Serial Schedule

The flooding schedule for message passing is in a parallel manner, i.e., all chip nodes update at the same time, then all variable nodes update simultaneously. In the case of cycle-free signature, the belief will converge to the exact a posteriori probability after a finite number of iterations that is bounded by half length of the longest path in the signature. Generally speaking, signature can not avoid cycle, and the propagated information may lead to an incorrect posterior probability [9]. In the flooding schedule, the updated message has to be stored until all the other nodes complete the new updating, which means the new message can not join the belief propagation immediately. Thus the convergence speed is slow and the detection performance is also limited. Furthermore, since all iterative messages are float-point numbers, high speed processors and large memory registers are required for hardware implementation.

In order to improve the convergence rate and reduce MUD complexity, we present a serial schedule for LDS-OFDM. In the serial schedule, the chip nodes update message sequentially. Unlike the flooding schedule where the new message can only be used in the next iteration, the serial schedule allows immediate propagation of new messages, and it is more efficient in terms of convergence rate and hardware cost. In the serial schedule, we use $L_{i,v}$ and $L_{i,v}^{j} \rightarrow r_{i,v}$ to compute $L_{i,v}^{j} \rightarrow r_{i,v}$ on the fly, avoiding additional memory to store $L_{i,v}^{j} \rightarrow r_{i,v}$. Such processing is derived by combining (10) and (8). It gives the following expression for the updating of chip node $c_v$:

$$L_{i,v}^{j} \rightarrow r_{i,v} = \kappa_{i,v} \max ( \sum_{(i') \neq i} L_{i',v}^{j-1} - \frac{1}{2\sigma^2} || y_v - r_i^{v} y_{i'}^{v} ||^2 )$$

(13)

$$L_{i,v}^{j} \rightarrow r_{i,v} = \kappa_{i,v} \max ( \sum_{(i') \neq i} (L_{i',v}^{j-1} - L_{i',v}^{j} \rightarrow r_{i,v}) - \frac{1}{2\sigma^2} || y_v - r_i^{v} y_{i'}^{v} ||^2 )$$

(14)

The detailed procedures are described in the sequel.

- Initialization

$$L_{i,v}^{j} \rightarrow r_{i,v} = 0, \forall k, \forall m, \forall n$$

(15)

$$L_{i,v}^{j} \rightarrow r_{i,v} = 0, \forall k, \forall m, \forall n$$

(16)

- Chip nodes updating

1) Accumulating all the messages delivered to the chip node $c_v$:

$$S = \sum_{(i') \in \varepsilon} (L_{i',v}^{j-1} - L_{i',v}^{j} \rightarrow r_{i,v})$$

(17)

2) For each variable node that is connected to the chip node $c_v$:

$$L_{i,v}^{j} = L_{i,v}^{j-1} - L_{i,v}^{j} \rightarrow r_{i,v}$$

(18)

$$L_{i,v}^{j} \rightarrow r_{i,v} = \kappa_{i,v} \max (S - L_{i,v}^{j} - L_{i,v}^{j} \rightarrow r_{i,v})$$

(19)

$$L_{i,v}^{j} = L_{i,v}^{j-1} + L_{i,v}^{j} \rightarrow r_{i,v}$$

(20)

3) Estimation

$$v_{i,v} = \arg \max_{v_{i,v}} L_{i,v} (v_{i,v})$$

(21)

Obviously, compared with flooding schedule, more fresh information can be utilized in the serial schedule. Thus the convergence rate and system performance can be improved, which will be shown in section V.

### IV. ITERATIVE LDPC DECODING

As the low density matrices in LDPC codes are very similar to the low density signature of LDS-OFDM, we utilize LDPC codes for LDS-OFDM system. Message passing algorithm, also referred as the sum product algorithm (SPA), is well-known for LDPC decoding [10][11]. The normalized min-sum (NMS) algorithm is a simplified version of SPA that can reduce computation complexity significantly without loss of decoding performance. The NMS performs belief propagation iteratively and outputs the a posteriori probabilities of the coded bit. Similar to the low density signature of LDS-OFDM, the LDPC code is based on a low density parity check matrix $H$ with dimensions $M \times N$. Each row in $H$ represents a parity check equation, while each column corresponds to a coded bit. Let $r_{v,m}$ be the LLR delivered from variable node to check node, $r_{c,v}$, be the LLR delivered from check node to variable node, and $R_{c,v}$ be the soft estimation of variable node. We present the NMS as follows.
- Initialization

The output of MUD, $L_{t,x}$, is sent to LDPC decoder as initial value.

- Parity check nodes updating

$$R_{v,x} = \alpha \times \beta$$  \hspace{1cm} (22)

where $\alpha = \text{sign}(R_{t,x})$ and $\beta = \text{abs}(R_{t,x})$, then

$$R_{v,x} = \prod_{\text{excluding self edges}} \alpha \times \beta$$  \hspace{1cm} (23)

- Variable nodes updating

$$R_{x,v} = L_{t,x} + \sum_{\text{excluding self edges}} R_{v,x}$$  \hspace{1cm} (24)

- Estimation

$$R_x = L_{t,x} + \sum_{\text{all edges}} R_{x,v}$$  \hspace{1cm} (25)

The LDPC decoder can make hard decision according to the $R_x$.

- Syndrome computing

If syndrome equals to zero or the decoder reaches the maximum number iterations, the decoding is terminated; otherwise continue the iterations again.

V. SIMULATIONS AND COMPARISONS

In this section, the bit error rate (BER) performance of LDS-OFDM is evaluated through Monte Carlo simulations over ITU Pedestrian Channel B. Simulation parameters are chosen as follows. The number of users is 10, the FFT size is 64, the sub-channel bandwidth is 15 KHz and the system overloading is 200%. The low density signature of LDS-OFDM has 60 chip nodes and 120 variable nodes.

For fair comparisons and to exhibit the difference between flooding and serial schedules of iterative MUD, we first evaluate uncoded LDS-OFDM system. Fig. 3 shows the BER results for several schemes of iterative MUD. As can be seen from the figure, at the first few MUD iterations (iteration of 1 or 3), the serial schedule attains much better performance than the flooding schedule. This is due to the fact that the updated message can participate in the belief propagation when serial schedule is applied. Consequently, it is possible to gather more accurate information and accelerate the convergence rate.

On the other hand, as the iterative process goes on (iteration of 6), the gap between flooding schedule and serial schedule becomes smaller. We can see that at the 6th iteration, the curves of two schedules almost overlap. This phenomenon informs that these two schedules will eventually converge to the same point. It is worth noting that the performance of the serial schedule with 3 iterations is very close to that of flooding schedule with 6 iterations.

One of the main advantages of the iterative receiver for LDS-OFDM is its ability to support high loads while maintaining the affordable complexity and satisfactory performance. For multiuser detection, the complexity of LDS-OFDM is $O((|x|^r) \times)\), which is much less than $O((|x|^\ell) \times)\) -- the complexity of the conventional MC-CDMA, where $x$ denotes the constellation alphabet. According to Fig. 3, the serial schedule can further reduce detection iteration and complexity with marginal performance loss. Therefore, the advantages of the serial schedule are: 1) although detection performance of the serial schedule is nearly the same to that of the flooding schedule when the number of MUD iterations is large enough (more than 6), however, in some applications where there is a constraint on the number of allowable/affordable iterations due to the hardware cost or other reasons, the serial schedule can achieve much better performance than the flooding schedule thanks to the faster convergence rate; 2) the register or memory requirement of the serial schedule is less than that of the flooding schedule, as the serial schedule saves the memory space for $L_{t,x}$. The disadvantage of the serial schedule is that it causes longer processing delay than the flooding schedule. These conclusions can be extended to the coded LDS-OFDM system.

Fig. 4 illustrates different FEC for coded LDS-OFDM, where the serial schedule with 3 iterations is chosen for the MUD. The compared FEC includes (2, 1, 7) convolutional code, (60, 30) LDPC code and (300, 150) LDPC code. The decoding algorithm for the convolutional code is the MAP algorithm which is originally proposed by Bahl, Cocke, Jelinek and Raviv (BCJR) [11]. The NMS algorithm presented in Secion IV is adopted for LDPC decoding, where the maximum iteration number is set to 5. In Fig.4, we can see that the performance of LDPC codes depends on the code length, and both LDPC codes outperform the convolutional code. In the medium to high SNR region, the (300, 150) LDPC code can attain about 0.2 dB gain over the (60, 30) LDPC code and about 0.5 dB gain over the convolutional code. Therefore, LDPC codes are more suitable for LDS-OFDM system than the convolutional code. This follows from the fact that both LDS-OFDM and LDPC code are based on sparse graph, and message passing algorithms can be efficiently applied for detection or decoding in LDPC coded LDS-OFDM systems.
VI. CONCLUSIONS

LDS-OFDM is a promising candidate for future mobile communications, but its MUD convergence rate is not good enough, and its receiver complexity is relatively high. In this paper, a serial schedule is developed for the iterative MUD in LDS-OFDM system. In the proposed serial schedule, updated message can be assimilated immediately, hence the convergence rate is improved significantly. Meanwhile, compared with the conventional flooding schedule, the iteration number can be saved and the receiver complexity can be reduced. Furthermore, due to the similar spare structure, LDPC codes are utilized for LDS-OFDM system. Numeric results show that, by choosing proper numbers of iteration for serial scheduled MUD and LDPC decoding, it is possible to attain a satisfactory performance with affordable receiver complexity.

ACKNOWLEDGMENT

This work was supported by the UK Engineering and Physical Science Research Council (EPSRC) and the fifth generation (5G) project.

REFERENCES