Fast and Area-efficient Sphere Decoding Using Look-ahead Search

Se-Hyeon Kang and In-Cheol Park†
System LSI Division, Samsung Electronics, Republic of Korea
†Dept. of Electrical Engineering and Computer Science, KAIST, Republic of Korea
shkang@ics.kaist.ac.kr, icpark@ee.kaist.ac.kr

Abstract— **Sphere decoding enables maximum likelihood (ML) detection with fairly low complexity in the MIMO wireless systems, but it takes hundreds cycles at low SNR environment. This paper proposes a fast decoding algorithm to reduce the decoding cycles using look-ahead search. Since the proposed decoding algorithm utilizes hardware resources to add other sub-trees into the search space successively, it helps not to go down into the sub-tree that has a small value at the root node but has large values at the child nodes. Scaling and enumeration techniques are also presented, which are effective in implementing the proposed sphere decoder. As a result, the proposed decoder saves about 30% decoding cycles at the cost of small hardware overhead compared to the conventional decoder.**

Index Terms—**MIMO detection, sphere decoder, enumeration**

I. Introduction

Wireless communication has continuously tried to increase throughput by achieving higher spectral efficiency and improving link reliability, since bandwidth is a limited resource. Multiple-input-multiple-output (MIMO) communication systems have recently attracted significant attention as a fascinating solution to increase system capacity and link reliability [1]. In rich-scattering environments, spatial multiplexing using multiple antennas can realize high spectral efficiency by utilizing multi-path propagation which was traditionally a pitfall for wireless communications. Thus the MIMO technique has been proposed as extensions to current wireless communication standards such as HSDPA and IEEE 802.11 and is part of emerging standards such as IEEE 802.16.

In MIMO systems, brute-force ML detection is infeasible for practical systems. Thus Zero-Forcing or V-BLAST is chosen as an alternative solution to trade off between BER performance and hardware complexity. Recently, sphere decoding algorithm has been proposed to limit the search space to a sphere centered at the received vector, leading to ML detection without exhaustive search [2]. The search space is represented by a tree where all the possible symbols for an antenna become the child nodes and prunes a sub-tree which has larger partial Euclidean distance (PED) than the radius of the current sphere. It is a kind of the depth-first search algorithm that finds a transmitted symbol associated with the minimum Euclidean distance to the received vector. Thus wrong initial choice causes redundant decoding cycles due to back-tracking of the tree. Large decoding cycles make it difficult to meet the high throughput requirement of the next generation communication standards.

There have been many researches to reduce the number of search cycles. The *k*-best algorithm is proposed to determine the transmitted symbol after a few constant cycles [3]. As inferred from the name, it keeps *k*-best candidates at every cycle while traversing the tree. As a result, the *k*-best algorithm does not search all the tree nodes and does not guarantee that the determined symbol has the minimum Euclidean distance. To efficiently find the closest lattice point, *Schnorr-Euchner* enumeration is used to start searching around the *Babai* point [4, 5]. It tells the next best node in the current sub-tree, but does not consider the nodes in the other sub-trees. Recent research shows a tendency to prune the search space aggressively by reducing the searching radius probabilistically at the higher level of the tree [6]. Sufficient experimental results may help choose an appropriate radius reduction ratio to save redundant decoding cycles, but it does not guarantee the ML solution.

In addition, some researches search the other sub-trees in parallel to overcome the disadvantage of the depth-first search [7]. To make these approaches meaningful, the additional hardware resources needed for searching the other sub-trees should be minimized. This paper proposes a new decoding algorithm that utilizes computation units to add the other sub-trees into the search space successively. The PED computation for the current sub-tree is replaced with simple enumeration. The comparison complexity remains the same as that of the conventional one. The proposed algorithm effectively searches 2 levels at once and increases the diversity order of choosing a sub-tree to be explored first, leading the ML solution in reduced cycles.

The rest of this paper is organized as follows. The MIMO system model and the sphere decoding algorithm is reviewed in Section II. Section III proposes a fast sphere decoding algorithm which reduces search cycles and explains two techniques to alleviate the hardware overhead. The performance of the proposed sphere decoder is summarized in Section IV. Finally concluding remarks are made in Section V.

II. Sphere Decoder

We consider a MIMO system with n_T transmit antennas and n_R receive antennas. It is assumed in this paper that square M-QAM constellation is used and the same constellation is employed for all the sub-streams.

A. System Model

Assuming narrow band transmission, a fading channel between a pair of transmit and receive antennas can be modeled

1550-2252/$25.00 ©2007 IEEE

as a flat one, resulting in an $n_T \times n_R$ complex-valued channel matrix C whose element c_{ij} represents the complex fading gain from the j-th transmit antenna to the i-th receive antenna. The channel is assumed to be estimated accurately at the receiver. Thus the input and output relation of the MIMO channel can be simply written as $\mathbf{r} = \mathbf{C}\cdot\mathbf{u} + \mathbf{w}$, where $\mathbf{w}$ is a zero-mean complex white Gaussian noise vector, $\mathbf{u}$ is a transmitted vector and $\mathbf{r}$ is a received vector. Each component of the transmitted vector is independently drawn from the same complex constellation. For notational convenience, we define $m = 2n_T$ and $n = 2n_R$. To obtain a real-valued representation of this MIMO system, we transform the complex-valued matrix equation into the real-valued matrix equation, $\mathbf{y} = \mathbf{H}\cdot\mathbf{x} + \mathbf{n}$, where

$$\mathrm{y} = \begin{bmatrix} \mathrm{Re}\{\mathrm{r}^T\} & \mathrm{Im}\{\mathrm{r}^T\} \end{bmatrix}^T \tag{1}$$

$$\mathrm{x} = \begin{bmatrix} \mathrm{Re}\{\mathrm{u}^T\} & \mathrm{Im}\{\mathrm{u}^T\} \end{bmatrix}^T \tag{2}$$

$$\mathrm{H} = \begin{bmatrix} \mathrm{Re}\{\mathrm{C}\} & -\mathrm{Im}\{\mathrm{C}\} \\ \mathrm{Im}\{\mathrm{C}\} & \mathrm{Re}\{\mathrm{C}\} \end{bmatrix} \tag{3}$$

$$\mathrm{n} = \begin{bmatrix} \mathrm{Re}\{\mathrm{w}^T\} & \mathrm{Im}\{\mathrm{w}^T\} \end{bmatrix}^T \tag{4}$$

B. *Sphere Decoding Algorithm*

The ML detection is to find a vector $\hat{\mathrm{x}}$ that has the smallest Euclidean distance to y as represented in Equation (5).

$$\hat{\mathrm{x}} = \arg\min_{\mathrm{x}\in S^m} \|\mathrm{y} - \mathrm{H}\cdot\mathrm{x}\| \tag{5}$$

In other words, a vector $\mathbf{x}$ can be regarded as a coordinate of the lattice, the sphere decoding is equivalent to searching a lattice point that is closest to a given point $\mathbf{y}$. All the Euclidean distances from all possible lattice points, however, can not be calculated if the number of antennas and the order of modulation become large. The sphere decoding algorithm efficiently solves this problem by limiting the searching within a sphere centered at the received vector $\mathbf{y}$.

The radius of the sphere can be represented by Euclidean distance between the received vector and a lattice point. In order to break the problem into sub-problems, it is useful to consider the QR factorization of matrix $\mathbf{H}$, $\mathbf{H} = \mathbf{QR}$, where $\mathbf{R}$ is an $m \times m$ upper triangular matrix and $\mathbf{Q}$ is an $n \times m$ unitary matrix. Thus the radius of the sphere can be represented as follows

$$\begin{aligned} r^2 &= \|\mathrm{y} - \mathrm{H}\cdot\mathrm{x}\| = \|\mathrm{y} - \mathrm{QR}\cdot\mathrm{x}\| = \left\|\mathrm{Q}^{\mathrm{H}}\cdot\mathrm{y} - \mathrm{R}\cdot\mathrm{x}\right\| \\ &= \sum_{i=1}^{m}\left(z_i - \sum_{j=i}^{m} r_{ij}x_j\right)^2 = \sum_{i=1}^{m}\mathrm{INC}_i^{\,2} \end{aligned} \tag{6}$$

, where $\mathbf{z} = \mathbf{Q}^{\mathrm{H}}\cdot\mathbf{y}$, z_i is an element of vector $\mathbf{z}$, and x_i and r_{ij} represent an element of vector $\mathbf{x}$ and matrix $\mathbf{R}$, respectively. Equation (6) can be re-written in an iterative form as shown in Equation (7).

$$\mathrm{PED}_i = \mathrm{PED}_{i+1} + \mathrm{INC}_i^{\,2} \tag{7}$$

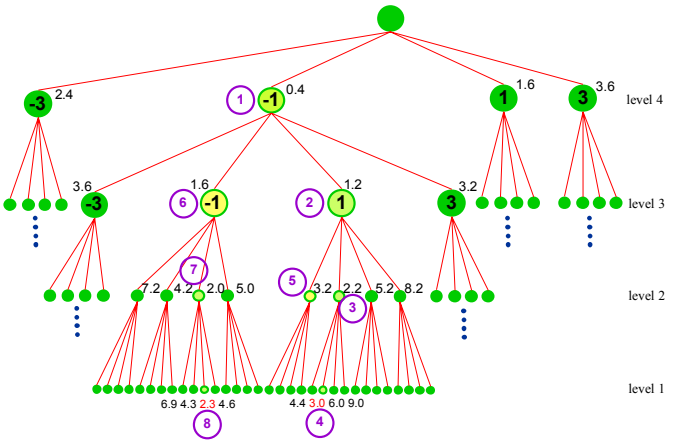

Figure 1. Depth-first search in sphere decoding algorithm

PED_i is the partial Euclidean distance of level i, which is a result of accumulating INCs from m to i. Thus PED_1 becomes the Euclidean distance. Since $\mathbf{R}$ is an upper triangular matrix, only x_i should be determined to get minimum PED_i at level i. x_j whose j is greater than i is already determined in the upper level. PED_i can be re-written by defining RCV_i as shown in Equation (8), where RCV_i can be computed by subtracting the effect of already determined symbols from z_i. To get INC_i, we have to select a symbol x_i that yields the smallest INC_i among the remaining symbols.

$$\begin{aligned} \mathrm{RCV}_i &\triangleq z_i - \sum_{j=i+1}^{m} r_{ij}x_j \\ \mathrm{PED}_i &= \mathrm{PED}_{i+1} + \left|\mathrm{RCV}_i - r_{ii}x_i\right|^2 \quad (x_i \in S) \end{aligned} \tag{8}$$

C. *Conventional Sphere Decoder*

Sphere decoding is analogous to the depth-first search as shown in Figure 1. All the possible values of x_i become sub-trees for x_{i+1}. At every level, it selects the best candidate for x_i. As traversing the tree, the partial Euclidean distance keeps increasing to the exact Euclidean distance of a lattice as indicated in Equation (7). Every time a valid lattice point is found, the search is restricted further by reducing the radius of

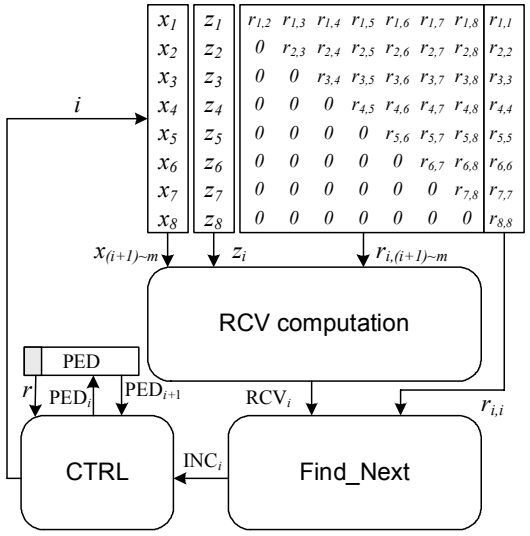

Figure 2. Conventional architecture of sphere decoder.

the sphere to the Euclidean distance of the leaf node.

The hardware architecture for the sphere decoding algorithm can be directly derived from the searching process. PED_i is evaluated for a node at every cycle. If it is larger than the current radius, it goes up one level. Otherwise, it goes down one level. The conventional sphere-decoding architecture for a 4×4 MIMO system is depicted in Figure 2. The rectangles in the upper side of Figure 2 are memories for storing the determined symbol x_i, the received symbol z_i and the channel coefficients r_{ij}. The RCV computation unit reads them to compute RCV_i. The Find_Next unit computes the minimum INC_i by selecting a symbol x_i among all the possible symbols of level i. The control unit adds the minimum INC_i to PED_{i+1} and compares it with the current radius to decide whether to go up or go down.

Since one node is evaluated at one cycle in the conventional sphere decoder, the number of visited nodes is equal to that of decoding cycles [8]. In low SNR environment, hundreds of decoding cycles are required. To implement a high speed sphere decoder, it is important to reduce the decoding cycles.

III. Proposed Sphere Decoder

This section describes a fast decoding algorithm to save redundant decoding cycles with small hardware overhead. Tracking-back in the dept-first search consumes too many cycles if the initial choice is wrong. The proposed algorithm enables to search the other sub-trees in parallel and this will increase the selection diversity at the cost of small hardware overhead.

A. Look-ahead Search

Figure 1 shows a part of the decoding tree for 2x2 16-QAM MIMO systems. Since the real-valued lattice is assumed, real and imaginary part of the symbol is computed at independent levels. Thus the number of levels is 4. At each level, there are 4 nodes which represent 4 possible symbols of 16 QAM. The order of visiting nodes in sphere decoding is indicated by the numbers in the circle. The value near each node represents its partial Euclidean distance. According to the depth-first search, the second node is selected in level 4 because it has the minimum PED. The second and third visited nodes are selected similarly as going down the tree. When it finds a leaf node at the fourth visit, the radius is set to 3.0 and goes up one level. The next visited node is the node which is right to the third visited node. Since the PED of the 5th node is larger than the current radius, it goes up one level again. The 6, 7 and 8-th nodes are determined in a similar way.

The depth-first search does not guarantee constant-cycle decoding. To prove that the current radius is the ML solution, the search process has to be continued till all the other nodes are pruned. If the initial choice is good enough to be a ML solution, other nodes are quickly pruned because they have larger PED's. Otherwise, much more nodes should be visited to find the ML solution. Initial choices, however, may be restricted to the child nodes of a node selected in the upper level. Since computation units are usually limited to $N = \sqrt{M}$, the other sub-trees can not be computed in parallel. If there is no restriction on the hardware resources, all the sub-trees can be computed at once to find the best candidate. No tracking-back is required in this case and hence the ML solution can be found after n cycles.

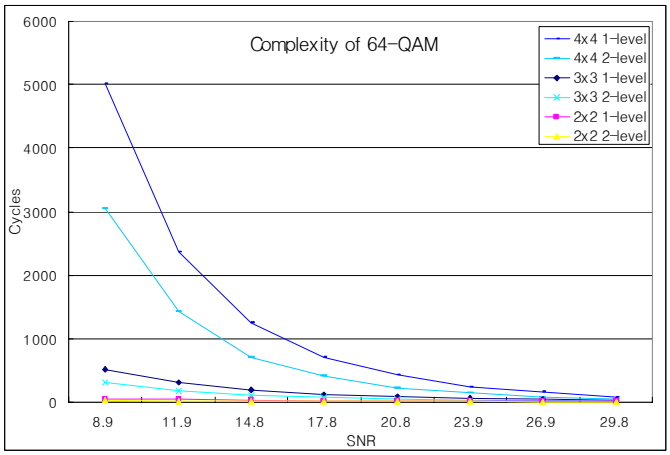

Figure 3. Number of decoding cycles for l-level and 2-level search

Let us suppose that a sphere decoder has N^2 computation units and the current node is the second visited node. As it can search 2 levels simultaneously, it selects the 7th node instead of 3rd one, saving redundant cycles. If we can find a smaller radius earlier, the more sub-trees can be pruned. The numbers of decoding cycles in different MIMO systems are depicted in Figure 3 for 1-level and 2-level searches, respectively. About 50% of decoding cycles is saved in 2-level search compared to the conventional 1-level search, but it is not an efficient method for reducing the decoding cycles because the hardware complexity is increased exponentially to N^2.

This paper proposes a look-ahead searching algorithm to efficiently increase the selection diversity. The proposed algorithm goes down by one level as the conventional one until the first leaf node is found. After that, it goes up and down by 2 levels, i.e. to even levels. Since the current sub-tree is already computed before, another sub-tree which is the next best candidate node in that level is computed by the existing hardware, which increases the hardware utilization.

Every time a level is visited, its sub-trees are added into the search space one after another as shown in Figure 4. Thus the other sub-trees are effectively looked up ahead in the current sub-tree. Figure 4 (a) represents the first visit when the search space is limited to one sub-tree as in conventional one; i.e. the second sub-tree in this case. At the second visit, the right sub-tree is computed and added to the search space as shown in Figure 4 (b). The shaded nodes represent the best candidate in each sub-tree. The *Schnorr-Euchner* enumeration determines the second sub-tree as the next best candidate at this time. Figure 4 (c) and (d) represent the third and fourth visit, respectively. After level i is visited N times, all the nodes whose parent node is the selected node in level i+1 are effectively added into the search space. Not all the nodes, however, are computed because the best candidate is determined by *Schnorr-Euchner* enumeration order within a sub-tree. Thus the comparisons are executed among the best candidates each

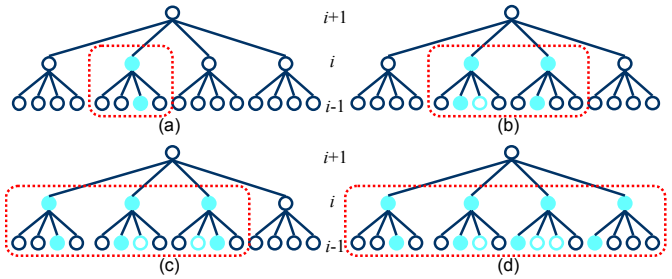

Figure 4. Concept of look-ahead search.

of which is selected for a sub-tree.

Sub-trees are added into the search space in *Schnorr-Euchner* enumeration order in level i+1 because the probability that a lattice point near the *Babai* point is the ML solution is higher than other points. If this assumption is true, the number of saved decoding cycles is close to that of 2-level search in Figure 3. The simulation results will be shown in Section IV.

B. Scaling and Enumeration

Two techniques are devised to reduce the hardware cost for the proposed algorithm. Since the proposed algorithm searches 2 levels simultaneously, N^2 PED's are compared maximally if implemented straightforwardly. Scaling and enumeration techniques will alleviate the comparison complexity.

The scaling technique is that the channel coefficients and the received vector elements are normalized by their diagonal elements as shown in Equation (9)

$$\mathrm{RCV}'_i \triangleq \frac{z_i}{r_{ii}} - \sum_{j=i+1}^{m} \frac{r_{ij}}{r_{ii}} x_j = z'_i - \sum_{j=i+1}^{m} r'_{ij} x_j \qquad (9)$$

$$\mathrm{PED}_i = \mathrm{PED}_{i+1} + r_{ii} \cdot \left|\mathrm{RCV}'_i - x_i\right|^2 \quad (x_i \in S)$$

where z'_i and r'_{ij} are the normalized received vector element and the normalized channel coefficient, respectively.

Thus the best candidate node can be easily found by looking up a few MSB bits of the RCV'_i. Comparison to find the minimum INC_i is not necessary. The corresponding diagonal coefficient is multiplied right before it is summed with PED_{i+1}. In the scaling technique, only one INC_i is calculated for the best candidate. To calculate INC_i for the other symbols, RCV_i is stored in memory. Next time level i is visited, INC_i for the next best candidate is computed by subtracting enumerated symbol x_i from the stored RCV_i.

To evaluate 2 levels in parallel, two enumeration units are required; one for level i and the other for level i-1. The first one is used to calculate INC_i in *Schnorr-Euchner* enumeration order. INC_{i-1} for the newly added sub-tree is computed by subtracting the best candidate from RCV_{i-1} computed by the original hardware. Another enumeration unit is used for the selected node in level i-1. Since only one node in level i-1 is selected among the node in all sub-trees, this enumeration unit is shared by all the sub-trees in level i-1. PED_i is calculated by adding the minimum sum of $\mathrm{INC}_i(x_i)$ and $\mathrm{INC}_{i-1}(x_i)$ to PED_{i+1} as expressed in Equation (10).

$$\mathrm{PED}_{i-1} = \mathrm{PED}_{i+1} + \min\left\{\sum_{j=i-1}^{i} \mathrm{INC}_j(x_i)\right\} \quad (x_i \in S) \qquad (10)$$

$\mathrm{INC}_i(x_i)$ is calculated by subtracting the enumerated symbol x_i from RCV_i. $\mathrm{INC}_{i-1}(x_i)$ is the value of the best candidate among the unvisited nodes whose parent is x_i. To find the minimum sum of INC_i and INC_{i-1}, a comparison unit capable of N input comparison at a time is required because the number of possible values for x_i is N. This can be the same hardware as the Find_Next unit of the conventional sphere decoder.

Enumeration unit performs the addition of the current symbol and delta. The delta is the amount of increment or decrement required for the current symbol to be the next best candidate. For example, if the *Schnorr-Euchner* order is 1, -1, 3 and -3, then the corresponding deltas are -2, 4 and -6. The absolute value of the delta is kept increasing by the difference of the neighbor symbols; i.e. 2 = 1–(-1) in this case. The sign is flipped every time till the resulting value goes beyond the symbol boundary. The proposed algorithm is summarized as follows.

```
Algorithm: LASD: x̂ = LASD(z, R)
1:  for(i=1; i≤n; i++) visit(i)=0;
2:  first = 1;
3:  Loop
4:     if(first) then
5:           RCV'_i = z'_i − Σ_{j=i+1}^{m} r'_ij x_j ;
6:           x_i = ENUM(visit(i), x_i, d_i, RCV'_i);
7:           INC_i = |RCV'_i − x_i|^2 ;
8:           PED_i = PED_{i+1} + r_ii · INC_i ;
9:           if(i != 1) then i--;
10:          else i++; first = 0;
11:          end if
12:    else
13:          x_i = ENUM(visit(i), x_i, d_i, RCV'_i);
14:          INC_i = |RCV'_i − x_i|^2 ;
15:          RCV'_{i-1} = z'_i − Σ_{j=i}^{m} r'_ij x_j ;
16:          x_{i-1} = ENUM(visit(i-1), x_{i-1}, d_{i-1}, RCV'_{i-1}
     );
17:          INC_{i-1} = |RCV'_{i-1} − x_{i-1}|^2 ;
18:          PED_{i-1} = PED_{i+1} + min{ Σ_{j=i-1}^{i} INC_j(x_i) } ;
19:          if( PED_{i-1} < r ) then
20:               if(i == 2) then
21:                    for(j=1; j≤n; j++) x̂_i = x_i ;
22:                    r = PED_{i-1} ;
23:                    visit(i)=0;
24:               else i -= 2;
25:               end if
26:          else
27:               if(i != n) then
28:                    visit(i)=0;
29:                    i += 2;
30:               else return x̂ ;
31:               end if
32:          end if
33:    end if
34: end Loop
```

Function: ENUM: x_i = **ENUM**(visit(i), x_i, d_i, RCV_i)

```
1:  if(visit(i) == 1) then
2:        x_i = x_i + d_i ;
3:        if( x_i < S_0 ) then
4:              visit(i)=2;
5:              d_i = Δ ;
6:              x_i = x_i + d_i ;
7:        else if( x_i > S_7 ) then
8:              visit(i)=2;
9:              d_i = -Δ ;
10:             x_i = x_i + d_i ;
11:       else  d_i = -sign(d_i)×(abs(d_i)+Δ) ;
12:       end if
13: else if(visit(i) == 2) then
14:       x_i = x_i + d_i ;
15: else
16:       visit(i)=1;
17:       x_i = [RCV_i]_M-Q ;
18:       d_i = (-1)^([RCV_i]_Q-1) · Δ ;
19: end if
```

※ DEFINITIONS: $(S_0, S_1, ..., S_7) = (-7, -5, ..., 7), \Delta = S_1 - S_0$

IV. PERFORMANCE EVALUATION

The architecture of the proposed sphere decoder is depicted in Figure 5. Since the scaling technique is applied, diagonal channel coefficients are fed into the control unit. The ENUM I unit calculates INC_{i+1}. The ENUM II unit executes enumeration for the selected x_i. The Find_Next unit selects the combination of x_i+1 and x_i which has the minimum sum of INC_{i+1} and INC_i. Compared with the conventional sphere decoder, one enumeration unit and the memories for RCV_{i+1} and RCV_i are additionally required. The decoding cycles, however, are reduced by about 30% compared to the conventional sphere decoding as shown in Figure 6. The simulation is performed for 3x3 and 4x4 64-QAM symbols. If the number of antennas or the order of modulation is increased, more decoding cycles will be saved.

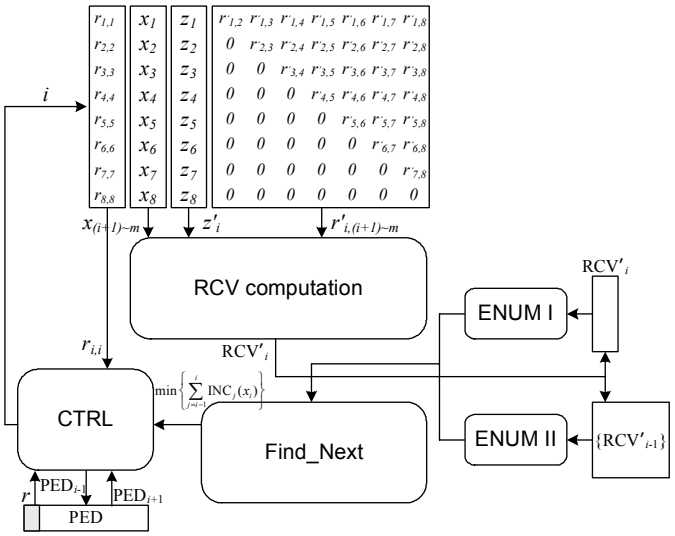

Figure 5. Architecture of the proposed sphere decoder.

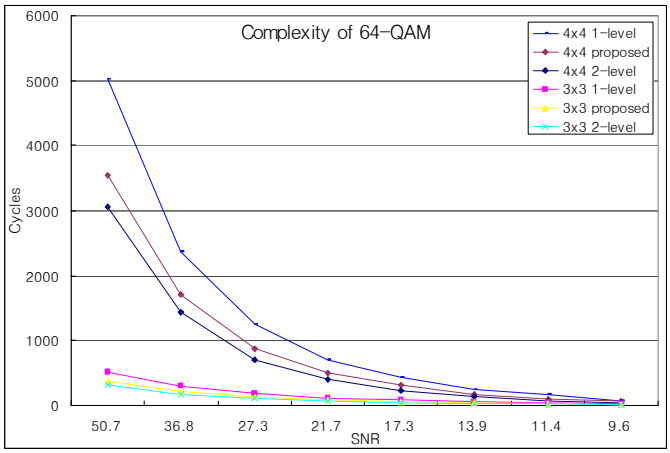

Figure 6. Comparison of decoding cycles.

V. CONCLUSION

In this paper, we have proposed a fast sphere decoding algorithm based on the look-ahead search to efficiently increase the selection diversity. The proposed algorithm goes up and down the tree by 2 levels. Whenever a level is re-visited, a sub-tree that is the next best candidate is added into search space. As the proposed sphere decoding algorithm can look ahead PED's of other sub-trees, it can make a better decision at the higher level. The scaling and enumeration techniques are also proposed to alleviate the hardware costs needed for the look-ahead search. Simulation results show that the proposed sphere decoding algorithm yields about 30% of decoding cycle reduction.

ACKNOWLEDGMENT

This work was supported by Institute of Information Technology Assessment through the ITRC and by IC Design Education Center (IDEC).

REFERENCES

[1] G. Foschini and M. Gans, "On the limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Commun.*, vol. 6, pp. 311-335, 1998.

[2] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels," *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 1639-1642, July 1999.

[3] K.-W. Wong, C.-Y. Tsui, R.S.-K. Cheng and W.H.Mow, "A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels," in *Proc. of ISCAS*, vol. 3, 2002, pp. 273-276.

[4] C.P.Schnorr and M.Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems," *Math. Programming*, vol. 66, pp. 181-191, 1994.

[5] M. O. Damen, H. E. Gamal, and G. Caire, "On maximum-liklihood detection and the search for the closest lattice point," *IEEE Trans. Inf. Theory*, vol. 49, no. 10, pp. 2389-2402, Oct. 2003.

[6] R. Gowaikar and B. Hassibi, "Efficient statistical pruning for maximum likelihood decoding," in *Proc. of ICASSP*, vol. 5, 2003, pp. 49-52.

[7] W. Xu, Y. Wang, Z. Zhou and J. Wang, "A Fast Exact ML Sphere Decoder with Efficient Two-layer Enumeration," in *Proc. of VTC* 2004 *Fall*, vol. 2, pp. 1309-1313.

[8] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner and H. Bölcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE Journal of Solid-State Circuits*, vol. 40, Jul. 2005, pp. 1566-1577.