Butterfly Log-MAP Decoding Algorithm

Jia Hou, Moon Ho Lee, and Chang Joo Kim

Abstract: **In this paper, a butterfly Log-MAP decoding algorithm for turbo code is proposed. Different from the conventional turbo decoder, we derived a generalized formula to calculate the log-likelihood ratio (LLR) and drew a modified butterfly states diagram in 8-states systematic turbo coded system. By comparing the complexity of conventional implementations, the proposed algorithm can efficiently reduce both the computations and work units without bit error ratio (BER) performance degradation.**

Index Terms: **E-function, Log-MAP, turbo code.**

I. INTRODUCTION

The iterative maximum a posteriori (MAP) decoder is a powerful tool for channel coding such as turbo code [1]–[3]. However, the implementation of the MAP decoder has high complexity. Therefore, simple and fast decoding algorithms had drawn much attention, e.g., [3]–[8]. On the other hand, it is well known that the fast Fourier transform (FFT) is a useful graphic butterfly algorithm to reduce the complexity and improve the speed of computation, especially in the multidimensional arrays [2]. Similarly, in this paper we generalized a modified states diagram for LLR computation in the systematic turbo decoder, which has a butterfly structure similar to the FFT. As a result, the LLR computations now consist of several simple butterfly units with different input information. By combining with a modified E-function, the serial and parallel architectures are designed in the following sections. In 3GPP standard, the 8-states systematic turbo code is defined for wireless communications and a 16-states case is used in satellite communications. In this paper, we mainly investigated the 8-states systematic turbo code as an example to explain our proposal. The numerical results demonstrate that the proposed butterfly Log-MAP decoding algorithm can efficiently reduce the number of work units and computations without BER performance degradation.

This paper is organized as follows. Section II first introduces the Log-MAP decoding algorithm and a modified E-function formula. Section III draws the butterfly states diagram and generalizes the LLR formula for 8-states systematic turbo decoder. Next, several work units and algorithms are implemented. The simulations and numerical results are shown in Section IV. Finally, we draw a conclusion and comment on the future research direction.

Manuscript received February 27, 2002; approved for publication by Sang Wu Kim, Division I Editor, May 11, 2004.

J. Hou and M. H. Lee are with the Institute of Information & Communication, Chonbuk National University, Chonju, 561-756, Korea, email: houjiastock@hotmail.com, moonho@chonbuk.ac.kr.

C. J. Kim is with the radio technology research group of ETRI, Korea, email: cjkim@etri.re.kr.

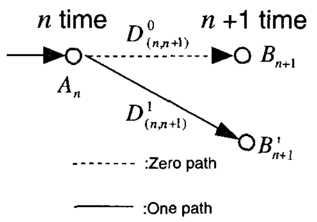


Fig. 1. Classical LLR function of turbo code.

II. LOG-MAP DECODING FOR SYSTEMATIC TURBO CODES

Normally, the Log-MAP decoding algorithm need to calculate all parameters of the metrics [1]. In a typical turbo decoder, the LLR computations are used to export the updated results after collecting the forward and backward information. The classical LLR function in turbo code can be plotted simply, as shown in Fig. 1. It can be mathematically written by

$$\begin{aligned}\mathrm{LLR}(d_n) = \log(\exp(A_n + D^0 + B_{n+1})) \\ - \log(\exp(A_n + D^1 + B'_{n+1})),\end{aligned} \tag{1}$$

where n denotes the time state, D^p is the metric coefficient with $p \in \{0,1\}$ on p path, A and B are forward and backward information, respectively.

Now let us consider an 8-states systematic turbo decoder in 3GPP standard. Its trellis diagram is shown in Fig. 2 [4], [6]. In this figure, four groups are formed according to the different branch values, where D presents the metric coefficient, $A(j)$ and $B(j)$ are forward and backward information at the jth trellis state, the solid lines and dashed lines denote the *zero*-path and *one*-path, respectively. By setting two adjacent trellis states to one metric group, which is noted as K, we can first calculate the information group by group with the same branch metric values, and finally generalize the LLR output of different branches by using the Log-MAP decoding algorithm. The log-function of $K = i$ metric group on p path from n time to $n+1$ time is given by

$$\begin{aligned}L(K^p = i) = \log(\exp(A_n + D^p + B_{n+1}) \\ + \exp(A'_n + D'^p + B'_{n+1})),\end{aligned} \tag{2}$$

where A', D', and B' are the coefficients coming from the other branch in the same metric group and the same path corresponding to A, D, and B. Collecting all metric groups, the output of LLR computation is obtained by

$$\mathrm{LLR} = L^0 - L^1 = L(K^0) - L(K^1), \tag{3}$$

where L^p is the log-likelihood on p path and $L(K^p)$ is the log-likelihood of all the groups K on p path.

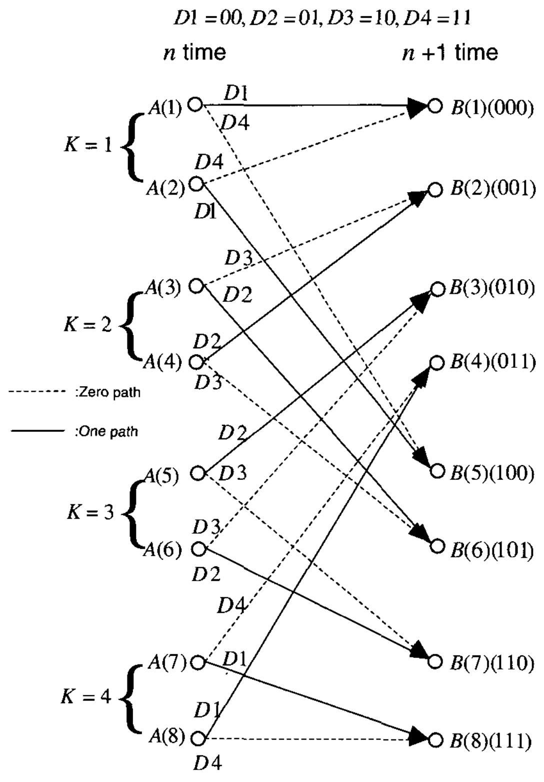


Fig. 2. The trellis diagram of 8-states systematic turbo code.

The detail formula of path of the 8-states systematical turbo code is given as

$$\begin{aligned} &L(K^p) \\ &= \log(\exp(\log(\exp(L(K^p=1))+\exp(L(K^p=4)))) \\ &\quad +\exp(\log(\exp(L(K^p=2))+\exp(L(K^p=3))))), \end{aligned} \tag{4}$$

where the branch metric value D in the groups of $\{K=1, K=4\}$ and that in the groups of $\{K=2, K=3\}$ are the same, as shown in Fig. 2. We now derive some useful equations for modified E-function as below.

A. The LLR Computation of Log-MAP Decoding Algorithm in Turbo Code [1]

For any trellis state, the LLR can be denoted by

$$\mathrm{LLR}_{anystate} = \log\frac{\exp(A_n + D^0 + B_{n+1})}{\exp(A_n + D^1 + B'_{n+1})}. \tag{5}$$

Generalizing all states in the trellis diagram, we have

$$\mathrm{LLR} = \log\frac{\sum\limits_{state=1}^{8}\exp(A_n(state)+D^0+B_{n+1}(endstate))}{\sum\limits_{state=1}^{8}\exp(A_n(state)+D^1+B'_{n+1}(endstate))}. \tag{6}$$

It calculates the difference between the sum of the states of *zero*-path values and the sum of the states of *one*-path values.

B. Modified LLR Computation and E-Function

We now decompose all trellis states to four groups, as shown in Fig. 2. For each group, the log-function is calculated as (2). Combined with (3) and (4), the grouped LLR is presented as

$$\begin{aligned} \mathrm{LLR} &= L^0 - L^1 = L(K^0) - L(K^1) \\ &= \log(\exp(\log(\exp(L(K^0=1))+\exp(L(K^0=4)))) \\ &\quad +\exp(\log(\exp(L(K^0=2))+\exp(L(K^0=3))))) \\ &\quad -\log(\exp(\log(\exp(L(K^1=1))+\exp(L(K^1=4)))) \\ &\quad +\exp(\log(\exp(L(K^1=2))+\exp(L(K^1=3))))) \\ &= \log\frac{\sum\limits_{K^0=1}^{4}\exp(L(K^0))}{\sum\limits_{K^1=1}^{4}\exp(L(K^1))}. \end{aligned} \tag{7}$$

Taking (2) into (7), easily we have

$$\begin{aligned} \mathrm{LLR} &= \log\frac{\exp(\log(\exp(A_n(1)+D4+B_{n+1}(5))+\cdots}{\exp(\log(\exp(A_n(1)+D1+B_{n+1}(1))+\cdots} \\ &= \log\frac{\exp(A_n(1)+D4+B_{n+1}(5))+\cdots}{\exp(A_n(1)+D1+B_{n+1}(1))+\cdots}. \end{aligned} \tag{8}$$

Consequently, this form can be generalized as

$$\mathrm{LLR} = \log\frac{\sum\limits_{state=1}^{8}\exp(A_n(state)+D^0+B_{n+1}(endstate))}{\sum\limits_{state=1}^{8}\exp(A_n(state)+D^1+B'_{n+1}(endstate))}. \tag{9}$$

Obviously, (9) is equivalent to the function (6). To conveniently implement the log-function, an E-function was proposed for Log-MAP decoder [7]. The mathematical model is defined as

$$\begin{aligned} &(A+D+B)E(A'+D'+B') \\ &\triangleq \log(\exp(A+D+B)+\exp(A'+D'+B')) \\ &\equiv \max((A+D+B),(A'+D'+B')) \\ &\quad +\log(1+\exp(-|(A+D+B)-(A'+D'+B')|)). \end{aligned} \tag{10}$$

Normally, the E-function is implemented by using a *look-up* table. We so call it *Jacobian log-function*, which is given by

$$aEb \equiv \max(a,b) + c\log(1+e^{-|a-b|/c}), \tag{11}$$

$$f(z) = c\log(1+e^{-|a-b|/c}), \tag{12}$$

and

$$c = A/Lc/mag(\sigma), \tag{13}$$

$$A = \frac{0.65(2^{q-1}-1)}{mag(\sigma)}, \tag{14}$$

where A is the no noise amplitude, q is the number of the transmitted bits, and signal-to-noise ratio (SNR) parameter $mag(\sigma)$ is obtained as

$$mag(\sigma) = \sigma\sqrt{\frac{2}{\pi}\exp\left(-\frac{1}{2\sigma^2}\right)+1-2Q\left(\frac{1}{\sigma}\right)}. \tag{15}$$

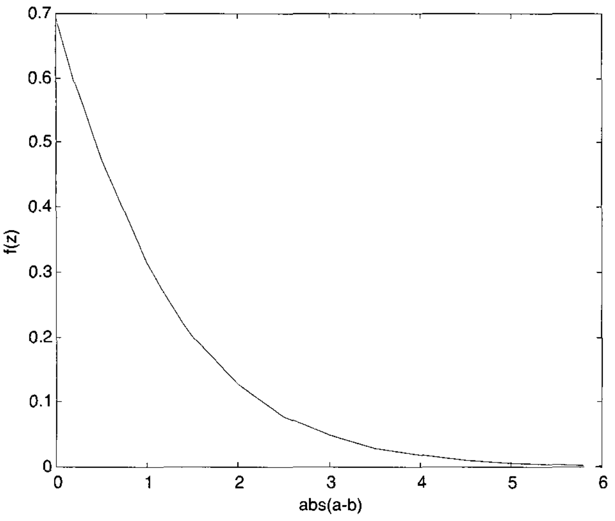


Fig. 3. *Jacobian log-function* with $c = 1$.

Table 1. The parameters of E-function for turbo code.

E_b/N_0[dB]	A	σ	Lc	$mag(\sigma)$	c
0.00	15	1.22	1.33	1.25	8.99
0.50	15	1.15	1.49	1.22	8.21
1.00	15	1.09	1.67	1.19	7.50
1.50	15	1.03	1.88	1.16	6.84

The numerical results and some related parameters of E-function are listed in Table 1. Usually, for turbo decoder, the log-function of *look-up* table $f(z)$ is shown as Fig. 3, where the parameter $c = 1$.

Especially, in the case of the 8-states systematical turbo code, we find that the metric value D is equal to D' if they are in the same metric group and on the same path, as shown in Fig. 2. Thus we can rewrite the E-function according to (4) and (10) as

$$\begin{aligned}
&(A+D+B)E(A'+D+B')\\
&= \log(\exp(A+D+B)+\exp(A'+D+B'))\\
&\equiv \max((A+D+B),(A'+D+B'))\\
&\quad +\log(1+\exp(-|(A+D+B)-(A'+D+B')|))\\
&= D+\max((A+B),(A'+B'))\\
&\quad +\log(1+\exp(-|(A+B)-(A'+B')|))\\
&\equiv (A+B)E(A'+B')+D, \qquad (16)
\end{aligned}$$

where the addition of the branch metric D in exponential is taken out, then the complexity of the computation is reduced apparently. In Fig. 4, the E-function is implemented by using only four values (A, B) and (A', B'), where LUT denotes the look-up table of the *Jacobian log-function.*

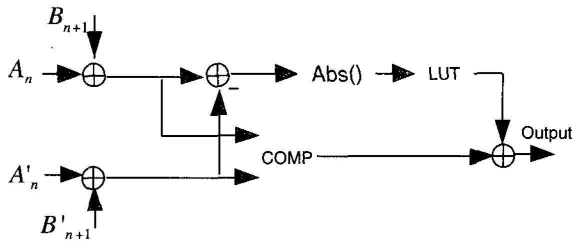


Fig. 4. Implementation of E-function.

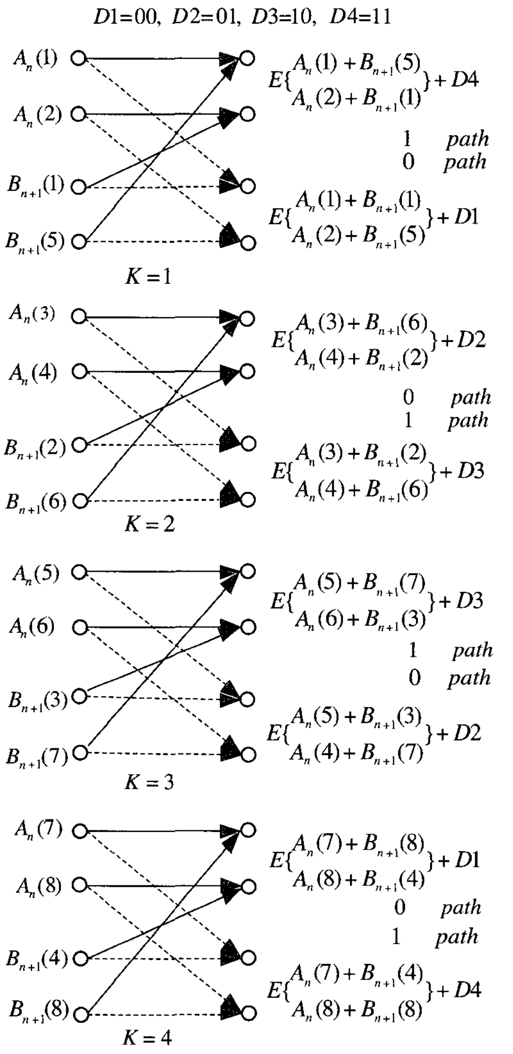

$D1$=00, $D2$=01, $D3$=10, $D4$=11
$A_n(1)$
$A_n(2)$
$B_{n+1}(1)$
$B_{n+1}(5)$
$E\{{A_n(1)+B_{n+1}(5) \atop A_n(2)+B_{n+1}(1)}\}+D4$
1 path
0 path
$E\{{A_n(1)+B_{n+1}(1) \atop A_n(2)+B_{n+1}(5)}\}+D1$
$K=1$
$A_n(3)$
$A_n(4)$
$B_{n+1}(2)$
$B_{n+1}(6)$
$E\{{A_n(3)+B_{n+1}(6) \atop A_n(4)+B_{n+1}(2)}\}+D2$
0 path
1 path
$E\{{A_n(3)+B_{n+1}(2) \atop A_n(4)+B_{n+1}(6)}\}+D3$
$K=2$
$A_n(5)$
$A_n(6)$
$B_{n+1}(3)$
$B_{n+1}(7)$
$E\{{A_n(5)+B_{n+1}(7) \atop A_n(6)+B_{n+1}(3)}\}+D3$
1 path
0 path
$E\{{A_n(5)+B_{n+1}(3) \atop A_n(4)+B_{n+1}(7)}\}+D2$
$K=3$
$A_n(7)$
$A_n(8)$
$B_{n+1}(4)$
$B_{n+1}(8)$
$E\{{A_n(7)+B_{n+1}(8) \atop A_n(8)+B_{n+1}(4)}\}+D1$
0 path
1 path
$E\{{A_n(7)+B_{n+1}(4) \atop A_n(8)+B_{n+1}(8)}\}+D4$
$K=4$

Fig. 5. Graphic algorithm for 8-states systematic turbo code.

III. PROPOSED BUTTERFLY LOG-MAP DECODING ALGORITHM

To achieve a high efficiency in hardware implementation, a better alternative is a joint design of modified E-function, generalized graphic formula and simplified architecture. It can be done by using the butterfly Log-MAP decoding algorithm as below.

As shown in Fig. 5, we propose an algorithm to calculate the E-function as the butterfly graphic processing. Since the structures of the computation equations for all groups are the same, we can derive a generalized pattern to denote these butterfly units, as shown in Fig. 6. The mathematical model is

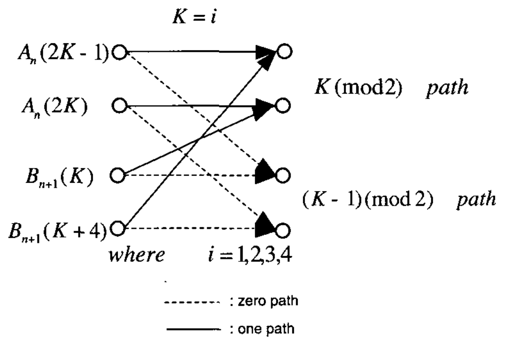


Fig. 6. Generalized trellis graph for 8-states systematic turbo code.

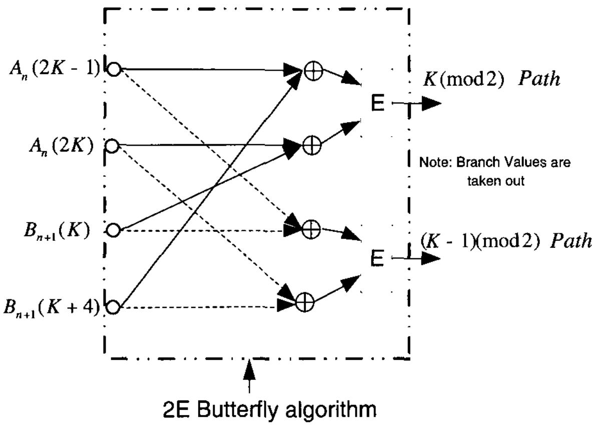


Fig. 7. Block diagram of the $2E$ butterfly algorithm.

$$\begin{aligned}(A_n(2K-1)+B_{n+1}(K+4))&E(A_n(2K)+B_{n+1}(K))\\ &+\left[\left(K+\left\lfloor\frac{K-2}{2}\right\rfloor\right)\bmod 2\right]\times D(5-K)\\ &+\left[\left(K-1+\left\lfloor\frac{K-2}{2}\right\rfloor\right)\bmod 2\right]\times D(K)\end{aligned}$$
on the [K mod 2] path, (17)

$$\begin{aligned}(A_n(2K-1)+B_{n+1}(K))&E(A_n(2K)+B_{n+1}(K+4))\\ &+\left[\left(K+\left\lfloor\frac{K-2}{2}\right\rfloor\right)\bmod 2\right]\times D(K)\\ &+\left[\left(K-1+\left\lfloor\frac{K-2}{2}\right\rfloor\right)\bmod 2\right]\times D(5-K)\end{aligned}$$
on the [(K − *1*) mod 2] path, (18)

where $\lfloor x\rfloor$ is the largest integer contained in x. In Fig. 6, we can observe that the generalized LLR computations are based on the Kth group and the modified E-function introduced in Section II. Therefore, the generalized graphic algorithm can be implemented as shown in Fig. 7. We call this unit as a $2E$ butterfly algorithm unit, which means that two E-Functions are used in this butterfly structure. In this implementation, we take the branch metric coefficient D out of the E-function as (16), and calculate the log-function on $\{0,1\}$ path, respectively. Thus we can write the output of the $2E$ butterfly algorithm unit as

$$(A_n(2K-1)+B_{n+1}(K+4))E(A_n(2K)+B_{n+1}(K))$$
on the [K mod 2] path, (19)

$$(A_n(2K-1)+B_{n+1}(K))E(A_n(2K)+B_{n+1}(K+4))$$
on the [(K − *1*) mod 2] path. (20)

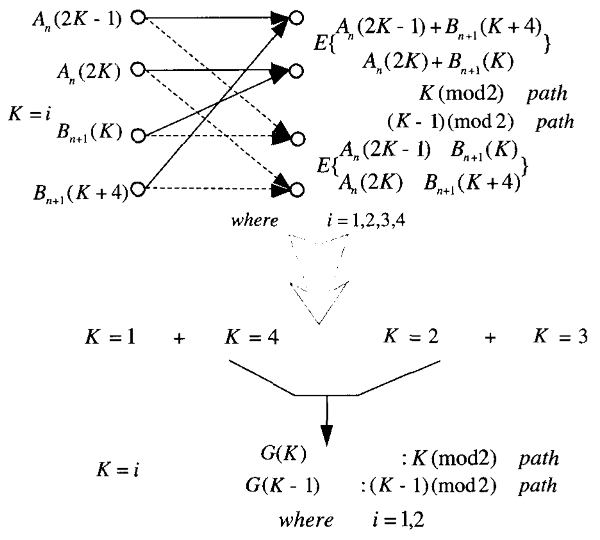


Fig. 8. The final generalized algorithm for 8-states systematic turbo code.

Since the trellis diagram of 8-states systematical turbo code is symmetric, we can easily generalize the groups further, as shown in Fig. 8.

The proposed formulas directly lead the algorithm to the final step, where the groups $\{K=1, K=4\}$ or the groups $\{K=2, K=3\}$ have the same branch values. The mathematical models are given by

$$\begin{aligned}G(K)=\{&(A_n(2K-1)+B_{n+1}(K+4))\\ &\times E(A_n(2K)+B_{n+1}(K))\}\\ &\times E\{(A_n(2(5-K)-1)+B_{n+1}(5-K))\\ &\times E(A_n(2(5-K))+B_{n+1}((5-K)+4))\}\\ &+(K(\text{mod }2)\times D(5-K)\\ &+(K-1)(\text{mod }2)\times D(K))\end{aligned}$$
on the [K mod 2] path, (21)

$$\begin{aligned}G(K+1)=\{&(A_n(2K-1)+B_{n+1}(K))\\ &\times E(A_n(2K)+B_{n+1}(K+4))\}\\ &\times E\{(A_n(2(5-K)-1)+B_{n+1}((5-K)+4))\\ &\times E(A_n(2(5-K))+B_{n+1}(5-K))\}\\ &+((K-1)(\text{mod }2)\times D(5-K))\\ &+K(\text{mod }2)\times D(K))\end{aligned}$$
on the [(K − *1*) mod 2] path. (22)

Obviously, in the final generalized formulas, two additions of the branch metric are skipped. Though the final structure seems

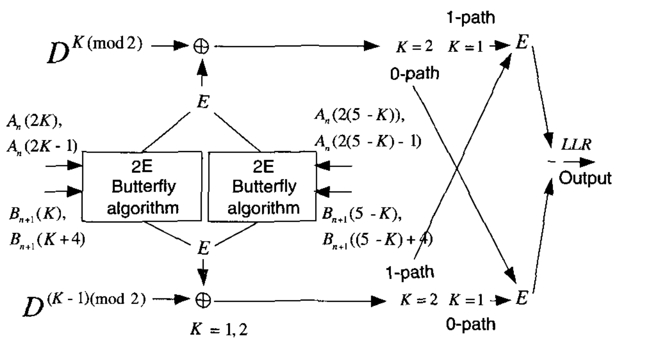


Fig. 9. The block diagram of final generalized butterfly algorithm for 8-states systematic turbo code.

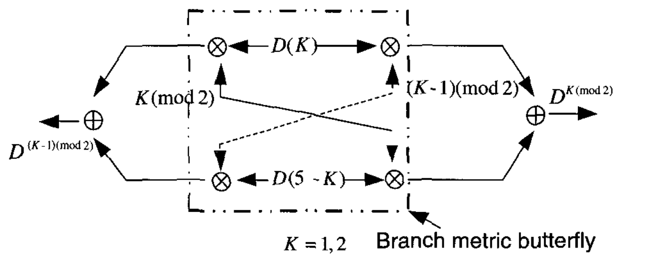


Fig. 10. The block diagram of the proposed branch metric value D computation.

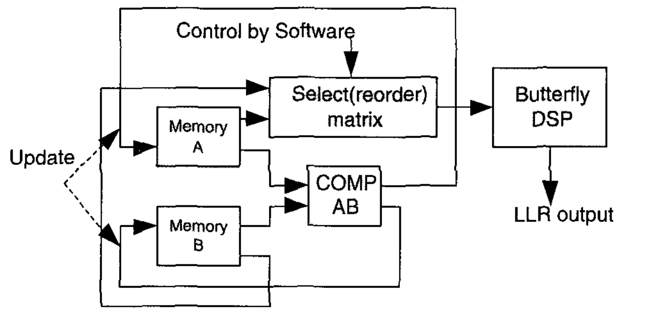


Fig. 11. The general block diagram of butterfly Log-MAP decoder.

to lead to little complexities than the previous introduction, in fact, it reduces the total number of computation and work units, since it generalizes two steps into only one. We implement the final generalized butterfly algorithm, as shown in Fig. 9, and design the calculation of the branch value D, as shown in Fig. 10. In these two figures, two additional butterfly structures are exploited to select the parameters from the different paths. As a result, there are so many butterfly structures in the proposed decoding algorithm. It is the reason why we call it butterfly Log-MAP decoding algorithm.

IV. SIMULATIONS AND NUMERICAL RESULTS

A general block diagram of butterfly Log-MAP decoder in 8-states systematical turbo code is shown in Fig. 11. To efficiently improve the performance of the Log-MAP decoder, in this section, the serial and parallel architecture are investigated.

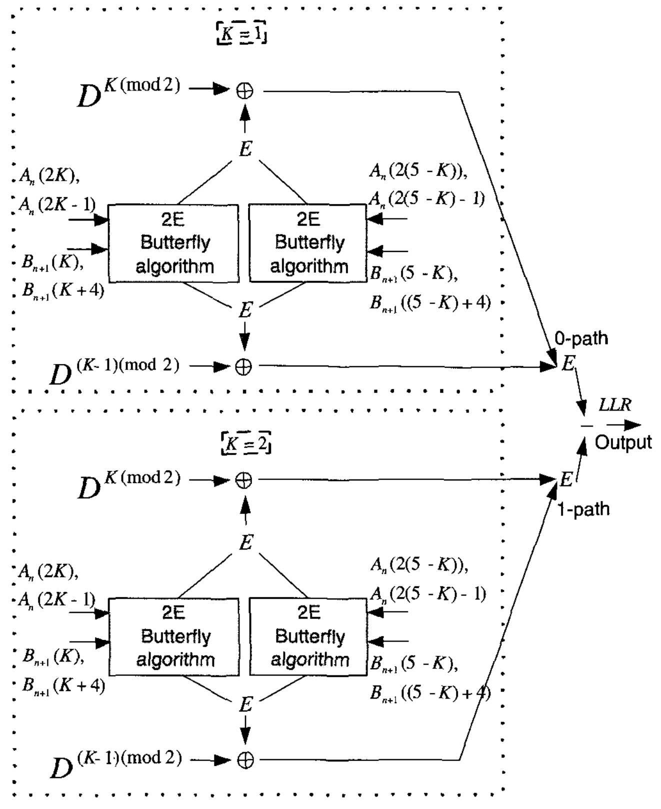


Fig. 12. The parallel butterfly Log-MAP decoder architecture.

A. Serial Butterfly Log-MAP Decoder Architecture

As mentioned in the above section, the same butterfly units are utilized for LLR computation, as shown in Fig. 9. Therefore, we can easily design a block diagram, in which we only need one butterfly unit, and the input information $\{A, B\}$ serially go through the calculator at different work units. Consequently, the serial architecture can use the least work units and complexity of the hardware. However, there is a demerit due to a longer time of delay.

B. Parallel Butterfly Log-MAP Decoder Architecture

To solve the time delay problem, the parallel butterfly Log-MAP decoder architecture is proposed, as shown in Fig. 12. The input information $\{A, B\}$ go through the calculator at the same time. However, there occurs a different demerit due to higher complexity than that of the serial case.

By comparing with the conventional parallel architecture, as shown in Fig. 13 [8], the proposed butterfly Log-MAP decoding algorithm offers a generalized butterfly pattern and a simple computation to realize the systematical turbo decoder system. Since the proposed algorithm is an equivalent transform from the conventional Log-MAP decoding, its BER performance is the same as that of the conventional case, as shown in Fig. 14.

Finally, the numerical results demonstrate that the serial butterfly architecture can reduce 81% additions from the conventional design, and save 25% simulation time, as shown in Table 2. In the case of parallel butterfly architecture, we find that the number of the additions is reduced about 60%. and the sim-

Table 2. VHDL design of the butterfly Log-MAP decoder for 8-states systematic turbo code.

	Conventional LLR decoder [8]	Serial butterfly architecture	Parallel butterfly architecture
The number of E-function	14	8	14
The additions in E-function	32	4	8
The outside additions for branch value	No	2	4
Total additions	32	6	12
VHDL compiler time	00:04:50	00:03:40	00:01:50
Assembler	00:00:02	00:00:02	00:00:02

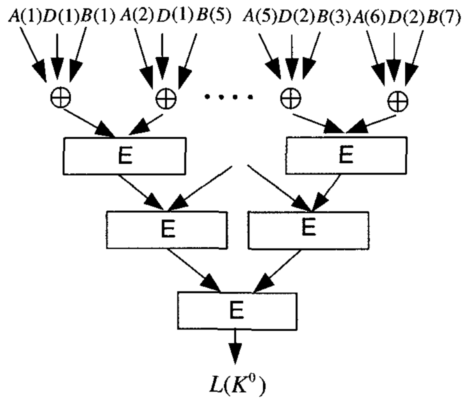


Fig. 13. The block diagram of the conventional parallel implementation scheme on 0-path [8].

ulation time is improved about 75% from the conventional decoder.

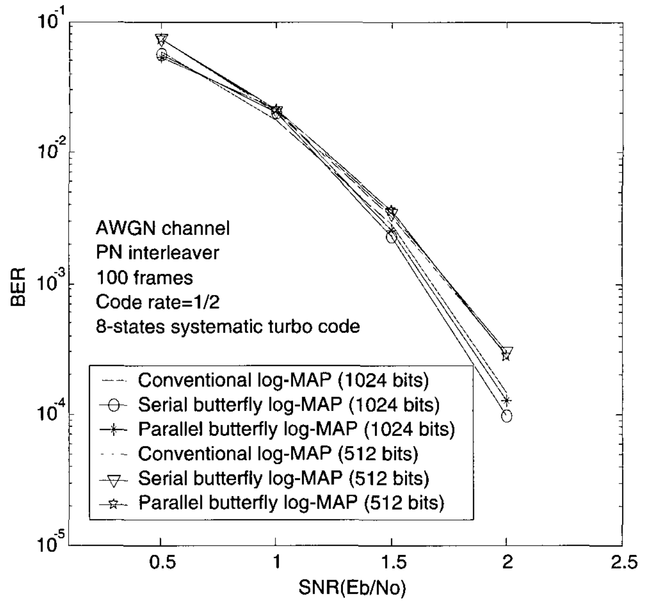


Fig. 14. The performance of the butterfly Log-MAP decoding algorithm for turbo code.

V. CONCLUSIONS

In this paper, a novel butterfly Log-MAP decoding algorithm for systematic turbo code is proposed. Different from the conventional turbo codes, we derived a generalized butterfly pattern to calculate the LLR in 8-states systematic turbo code system. By comparing the complexity of the conventional implementations, the proposed algorithm can efficiently reduce both the operations of computation and work units without BER performance degradation. In general, with the number of the states increasing, the proposed algorithm can approach substantial improvement from the conventional design. The proposed algorithm can be applied to other trellis codes, such as TCM and turbo-TCM, similarly.

ACKNOWLEDGMENT

This work was supported in part by Univ. IT Research Center Project, Ministry of Information & Communication, Korea.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding - turbo codes," in *Proc. ICC'93*, June, 1993, pp. 1064–1070.

[2] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 3, pp. 429–445, Mar. 1996.

[3] S. Y. Kim, J. Chang, and M. H. Lee, "Simple iterative decoding stop criterion for wireless packet transmission," *IEE Electron. Lett.*, vol. 36, no. 24, pp. 2034–2035, Nov. 2000.

[4] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and suboptimal MAP decoding algorithms operating in the log domain," in *Proc. ICC'95*, June, 1995, pp. 1009-1013.

[5] A. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes," *IEEE J. Select. Areas Commun.*, vol. 27, no. 2, pp. 260–264, 1998.

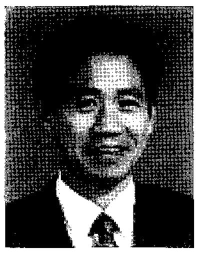
[6] J. Hagenauer, P. Robertson, and L. Papke, "Iterative ("turbo") decoding of systematic convolutional codes with MAP and SOVA algorithms," in *Proc. ITG'94*, Munchen, Germany, Oct. 1994, pp. 21–29.

[7] S. Steven, "Implementation and performance of a serial MAP decoder for use in an iterative turbo decoder," in *Proc. ISIT'95*, Apr. 1995, p. 471.

[8] E. Yeo *et al.*, "VLSI architectures for iterative decoders in magnetic recording channels," *IEEE Trans. Magn.*, vol. 37, no. 2, pp. 1024–1036, Mar. 2001.

[9] M. H. Lee, "High speed multidimensional systolic arrays for discrete Fourier transform," *IEEE Trans. Circuits Syst. II*, vol. 39, no. 12, pp. 876–879, Dec. 1992.

Jia Hou received his B.S. degree in Communication Engineering from Wuhan University of Technology in 2000, China, and M.S. degree in Information & Communication from Chonbuk National University in 2002, Korea. He is now a Ph.D. candidate at the Institute of Information & Communication, Chonbuk National University, Korea. His main research interests are sequences, CDMA mobile communication systems, error coding, and space time signal processing.



Moon Ho Lee received his B.S. and M.S. degrees both in Electrical Engineering, from the Chonbuk National University, Korea, in 1967 and 1976, respectively, and Ph.D. degree in electronics engineering from the Chonnam National University in 1984 and the University of Tokyo, Japan, in 1990. From 1970 to 1980, he was a chief engineer with Namyang Moonhwa Broadcasting. Since 1980, he has been a professor in the Department of Information and Communication and a director at the Institute of Information and Communication, both at Chonbuk National University. From 1985 to 1986, he was also with the University of Minnesota, as a Postdoctoral Fellow. He has held visiting positions at the University of Hannover, Germany, in 1990, the University of Aachen, Germany, in 1992 and 1996, and the University of Munich, Germany, in 1998. He has authored 20 books, including Digital Communication (Korea: Youngil, 1999), Information and Coding (Korea: Bokdu, 1998), Digital Image Processing, (Korea: Daeyoung, 1994), and Digital Filter Design (Korea: Daeyoung, 1995). His research interests include multidimensional source and channel coding, mobile communication, and image processing. He is a Registered Telecommunication Professional Engineer and a member of the National Academy of Engineering in Korea. He was the recipient of the paper prize award from the Korean Institute of Communication Science in 1986 as well as in 1987, the Korea Institute of Electronics Engineers in 1987, Chonbuk Province in 1992, and the commendation of the Prime Minister for inventing the Jacket matrix, in Korea, 2002.

Chang Joo Kim received B.S. degree in avionics engineering from Hankuk Aviation University in 1980, and M.S. and Ph.D. degrees in electrical engineering from KAIST, in 1988 and 1993, respectively. From 1980 to 1982 he was engaged as a research engineer at ADD. Since 1983 he has been with the communication fields of ETRI, where he is now a director of radio technology research group. His interests involve wireless communications and radio technologies.