Deadline Dependent Coding - A Framework for Wireless Real-Time Communication

Elisabeth Uhlemann and Per-Arne Wiberg
Halmstad University, SE-301 18 Halmstad,
http://www.hh.se, {elisabeth.uhlemann, per-arne.wiberg}@ide.hh.se

***Abstract* -- A framework for real-time traffic over a wireless channel is proposed in this paper. The deadline dependent coding (DDC) scheme makes use of modern tools from the information theory. A combination of hybrid ARQ and soft decision decoding is used to maximise the probability of delivering information before a deadline. The strategy of DDC is to combine different coding and decoding methods with ARQ in order to fulfil the application requirements. These requirements are formulated as two QoS parameters: deadline (t_{DL}) and probability for delivery before this deadline (P_d). The application can negotiate these parameters with the DDC protocol, thus creating a flexible scheme. It is still possible to make probabilistic guarantees on the communication mechanism.**

***Index Terms* -- real-time, communication**

I. INTRODUCTION

There is a tremendous development in the wireless communication field. New technologies and products reach the market at increasing rate. Some parts of this evolution look very promising for industrial applications.
Cabling always caused the industry problems in terms of costs and feasibility in general. Some applications even demand a wireless access in order to function.
There have been some implementations of wireless communication systems for industrial use. The problem of guaranteeing real-time delivery has been solved in quite an ad hoc manner. The consequence is that it is hard or impossible to make any judgement of the security aspects.
In this paper we will describe a framework of working with such unsecure transfer mechanisms as radio channels.
A real-time communication channel is characterised by the fact that it is equally important to deliver in time, as it is to deliver the correct data. In literature the real-time field often mentions two classes of real-time systems; hard and soft real-time systems. A hard real-time system is a system where you cannot tolerate any late delivery of a result. In a soft system on the other hand you can tolerate a late delivery but with a degraded value to the system.
The uncertainty concerning security in wireless transmission has prevented it from being used in hard real-time systems.
If we have control over the communication channel security, wireless communication can replace cables in a vast number of applications. A class of industrial applications is measurement and control on rotating parts. Here it is obvious why wireless communication is necessary.
Another application is communication to and from different kind of vehicles in factory automation situations. In control systems in general, cabling is something we would like to replace.
In all these cases it is not clear if it is a hard – or a soft real-time system. In general we would like to turn away from the notion of hard and soft real-time systems as if there only exist these two classes. In our framework we introduce a probabilistic view of the communication mechanism. This means it is no longer meaningful to talk about hard or soft real-time systems. We rather talk about deadline for delivery and the probability to succeed in delivering before this deadline. We therefore introduce two parameters: deadline (t_{DL}) and probability for delivery (P_d). These two parameters can be used for setting bounds on the communication system, or stating requirements.
The values of these parameters can be derived by means of classical safety analysis methods. In the process of making requirement analysis of the system t_{DL} and P_d is one result. If this shall be a powerful tool in designing real-time systems including wireless communication links, there must be mechanism in the protocol for guaranteeing the parameters.
If the two parameters are viewed as quality of service parameters (QoS) of a communication mechanism it means that a protocol layer can negotiate the parameters with an underlying layer. If we give the protocol these properties the communication system can guarantee the delivery based on the parameters or reject the transmission request. If the request is rejected the application have a possibility through an exemption.
What we strive for is to maximise the probability that the communication system will be able to accept the transmission request. Aspects that must be considered is not only static and dynamic channel properties like throughput and error rate, but also multi-user interference.
In communication systems one can use several means for increasing the probability of success. Different diversities are such means. Time diversity is one and frequency diversity is another. Both these tools are used in our approach to guarantee t_{DL} and P_d.
The main idea is that the t_{DL} and P_d parameters are mapped onto a retransmission protocol. The retransmission protocol

plays the role of maximising the success rate and still being able to reject requests that cannot be handled.
The protocol performs a series of transmissions with different amount of redundant information, thus the closer to the deadline the more information the receiver will have in the decoding process. We denote this series of transmissions a *transmission suite*.
In a multi-user radio system it is important to keep the interference, from all acting nodes, down. This is one of the reasons for trying to get the information across with as little redundant information as possible, thus the lower amount of redundant bits in the beginning of the transmission suite. Closer to deadline it is more important to get the information across than not interfering with others; so more redundant information is needed.
In section II the background and the main ideas behind the DDC scheme are described. A previous work by the authors [1] evaluating a DDC scheme based on hard decision decoding is discussed.
In this work a decoder based on a soft decision mechanism will be evaluated. Different decoding techniques and algorithms using soft decision decoding are examined in section III. The application of these algorithms to the DDC scheme is discussed.

II. Background

A node communicating in real-time must not only remit the correct information, but also deliver it within a certain limit of time, the deadline. As a missed deadline can have disastrous consequences, a real-time system must leave some sort of performance guarantee, a quality of service. When a hard real-time system is designed it is often assumed that the deadline must be met with unity probability. This is a situation that cannot be achieved with any physical system. As the consequences for missing a deadline is disastrous in a hard real-time system we want to quantify the probability for this event.
The application designer states the probability as a requirement. The communication mechanism then transforms this request into actions in the protocol, using models of the communication channel and different communication methods.
We base our system on the Code Division Multiple Access (CDMA) method, the Deadline Dependent Coding (DDC) scheme, and linear block codes as explained below.

A. Media Access Method

Code Division Multiple Access (CDMA) uses orthogonal codes in order to separate the different users and thereby providing multiple access.
The advantage of CDMA in our case is that it rejects multipath reception to some extent. Multipath reception is a typical phenomenon in radio communication and is the result of the message bouncing of obstacles and having to travel different paths of different length. Thus, multiple copies of the same message are received within a certain time window. The radio channel also experiences fading, usually with a Rayleigh distribution. By spreading the messages to be sent, by means of orthogonal codes, they use a wider frequency spectrum, thus reducing the fading damages on a particular frequency band. CDMA also gives instant access to the media, an important issue in a real-time system, where time is a valuable resource. Less administration is required when a new user is added as opposed to TDMA for example. Naturally here is a limit to the number of simultaneous users in a CDMA system, but it still degenerates gracefully.

B. Deadline Dependent Coding

In [1] the Deadline Dependent Coding (DDC) scheme was used. This coding scheme strives to meet the demands on deadline (t_{DL}) and probability of a correct delivery before the deadline (P_d), and at the same time keeping the activity factor of the network as low as possible. This is realised by using hybrid ARQ [2], which means that both an error correcting code and a retransmission scheme is used as opposed to just an error detecting code and retransmissions. The message to be transmitted is coded differently depending on the deadline and the requested delivery probability. In the beginning of the time window a high rate coded message, i.e. few redundant bits, is transmitted, see Fig. 1. If no acknowledge signal is received, the procedure will be repeated until a certain retransmission deadline is reached. When this occurs the receiver performs a bit-wise majority voting procedure in order to restore the correct message. If the receiver still fails to decode the result from the majority voting procedure a low rate coded message, i.e. a large number of redundant bits, will be sent as a last attempt to get a correct delivery before deadline.

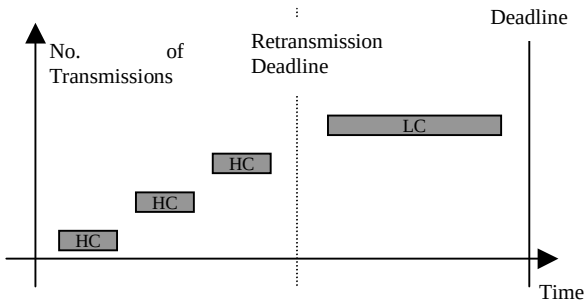

Fig. 1. The Deadline Dependent Coding (DDC) Transmission Suite.

C. Linear Block Codes

The error correcting codes used in the DDC scheme are Reed-Solomon codes. These codes are especially good at correcting burst errors, which is a common error type in a wireless system due to the fading.
Block codes introduce controlled amounts of redundancy into a transmitted data stream, providing the receiver with the ability to detect and correct errors caused by noise on

the communication channel. The data stream is divided into blocks and redundancy is added to each block independently, thus producing a larger block.
Each code can be described by a generator matrix, which is multiplied with the data bits to produce a code word, and a parity check matrix, which when multiplied with the received code word is zero if no errors are present.
The data source generates blocks of k messages symbols taking values from the Galois Field, GF(q) [2]. Each message block is then encoded, generating a code word of n code word symbols, each symbol taking values from GF(q). Consequently, the total amount of redundancy, r, introduced is $r = n - k$. The code word symbols are then sent to the modulator. We have used a BPSK modulation technique and as Reed-Solomon codes are nonbinary, the q–ary code word symbols must be translated into binary channel symbols before transmission. The receiver recovers the channel symbols from the demodulator and passes them along to the translator for conversion back to q-ary code word symbols. The received code word symbols are sent in n-symbol blocks to the error control decoder. As the noisy channel corrupts the modulated carrier, the symbol block arriving at the error control decoder contains errors.

D. Hard Decision Decoding

To decode the Reed-Solomon codes a bounded distance decoder was used in [1].
A bounded distance decoder will select the code word closest in *hamming* distance [2] to the received code word if and only if that distance is less than the bounded distance. If there is no code word within the bounded distance a decoder failure is declared. The hamming distance between two code words is the number of positions in which they differ. A bounded distance decoder will make an error whenever the received code word is within the bounded distance of another code word than that which was sent, see *Fig. 2*.

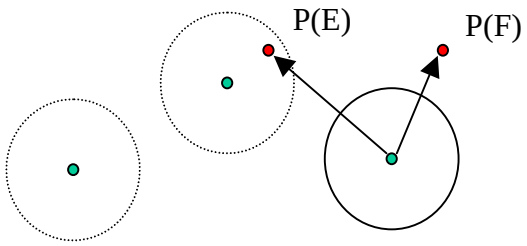

Fig. 2. Bounded Distance Decoding. In the centre of each sphere is a code word. P(F) is the probability of decoder failure and P(E) the probability of decoder error.

The hamming distance implies hard decision decoding, i.e. the decoder first determine whether it is a binary one or a zero in a particular position of the received code word by means of quantization and thereafter calculates the code word distance based on the binary representation of the code word. Decisions based directly on the unquantized demodulator output, so-called soft decisions decoding, requires a more complex decoder that can handle analog inputs, but offers a significant performance improvement over hard decision decoding.

III. Soft Decision Decoding of Linear Block Codes

Soft decision decoding is now introduced in the DDC scheme in order to enhance the performance with respect to probability of delivering before the deadline.
We have developed a toolbox with different decoding methods which can used in the DDC framework. These methods are based on a trellis representation of block code. The trellis concept is explained in section III.A. Two different decoding algorithms, the Viterbi and the BCJR algorithm, are explained in section III.B and III.C respectively. Both these decoding algorithms are used in conjunction with the trellis. The Viterbi algorithm has a lower complexity and is thus faster. The BCJR algorithm on the other hand can be used in a turbo decoding scheme, explained in section III.D, resulting in a faster and more powerful decoding strategy. Turbo decoding, which is an iterative decoding strategy, used in conjunction with ARQ gives new possibilities for adapting the transmission suite and the coding to a particular t_{DL} and P_d request.
Which strategy to use in which situation to make the most of the remaining time is what is to be examined and simulated in the near future.

A. Trellis Representation of Linear Block Codes

For many years the algebraic decoding of linear block codes was considered the only possible. Convolutional codes on the other hand could be decoded using soft decision decoding with the Viterbi algorithm on a trellis. The fact that block codes also can be interpreted as trellises was first described in [3] and later in [4].
Consider a binary (n,k) block code C over $GF(q)$ with parity check matrix, H. A message of k information bits is shifted into the encoder and is encoded into a code word of n code bits [5]. The code words in C are all the n-tuples $\mathbf{x}$ with elements from $GF(q)$, such that $\mathbf{Hx}=\mathbf{0}$. One n-bit code word is shifted out on the channel each interval t. This procedure can be viewed as a finite state machine, FSM, and C can be represented by an n-section trellis diagram of length t. A trellis is a directed graph consisting of n+1 states and branches, see Fig. 3.
A branch in the i-th section of the trellis connects a state $S_{i-1} \in \sum_{i-1}(C)$ to a state $S_i \in \sum_i(C)$ and is labeled with a code bit u_i that represents the encoder output at the bit interval from time (i-1) to time i. A branch represents a state transition. The branches diverging from the same state have different labels. Each path from the initial state S_1 to the final state S_t is a code word in C.

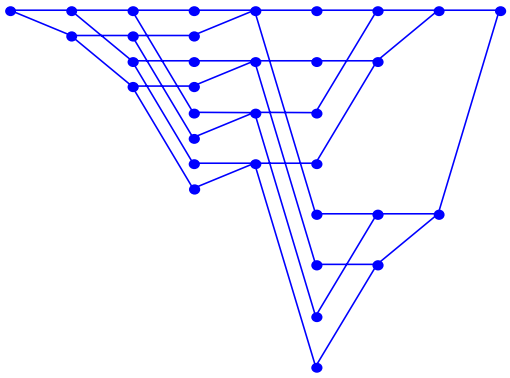

Fig. 3. Trellis for RM(8,4). Horizontal branches represent the label binary zero.

The collection of nodes at time *i* is obtained from the nodes at time (*i*-1) by:

$$S_i(l) = S_{i-1}(j) + \alpha_m \mathbf{h}_i; \quad m = 0,1,...,q-1, \quad \forall j \in \sum_i(C) \tag{1}$$

The transitions between the nodes at time *i*-1 and *i* are labeled by the particular value of $u_i = \alpha_m$.

The trellis representation described can be used for soft decision decoding of the block codes in the DDC scheme described in section II.C.

B. *Decoding with Viterbi*

A trellis can be used to perform soft decision decoding based, no longer on the hamming distance between code words, but the Euclidean distance. We simply search trough the trellis looking for the sequence that minimizes the Euclidean distance. However, we do not have to compute the distance for all q^k possible sequences. We can use the Viterbi algorithm [6] to eliminate certain sequences that merge in the trellis with other sequences that are much closer to the received sequence. The Viterbi algorithm works as follows:

Step 1) For block codes we know that we start at time *t*=0 in the all zero state, so we initialize its metric to zero. We then calculate the branch metric for the branches emerging from the all zero state. Either Euclidean metric or hamming metric may be used. The paths or survivors and their metric are stored for each new state.

Step 2) Increase *t* by one. Compute the metric for all the paths entering a state by computing the new branch metric and adding that to the corresponding state metric of the surviving state from the previous time step. For each step with a merger, store the path with the largest metric, the survivor, and its metric and discard all other paths.

Step 3) If *t* is smaller than the length of the trellis repeat step 2, otherwise choose the all zero state as "winner" and choose the corresponding sequence of survivors as output sequence. It should be noted that it is only for block codes that we know that we are in the all zero state in the end, otherwise it would not be a code word.

C. *Decoding with BCJR*

The Viterbi algorithm is optimal in the sense that it minimizes the word error probability by selecting the maximum likelihood sequence (called maximum likelihood sequence detection, MLSD). It does not, however, minimize the symbol or bit error probability, by choosing in each case the symbol or bit with the maximum aposteriori probability (MAP). The complexity of a MAP decoder is significantly higher and is, in general, not an alternative to MLSD since in most applications a MLSD decoder and a MAP decoder will have virtually identical performance. The main asset of the MAP decoder is that it produces aposteriori information bit probabilities, which can be used, in an iterative or concatenated decoding scheme.

An algorithm that does perform MAP decoding is the BCJR-algorithm [3]. The BCJR algorithm calculates the aposteriori probabilities for each symbol using the trellis as described below.

We define

$$\alpha_t(m) = \Pr\{S_t = m; Y_1^t\} \tag{2}$$

as the joint probability of the partial received sequence $(Y_1,...,Y_{t-1})$ and the state $S_t = m$

$$\beta_t(m) = \Pr\{Y_{t+1}^{\tau}, S_t = m\} \tag{3}$$

as the conditional probability of the remaining sequence $(Y_{t+1},...,Y_{\tau})$ excluding *t* given the state at time *t* is $S_t = m$; and

$$\gamma_t(m',m) = \Pr\{S_t = m; Y_t \mid S_{t-1} = m'\} \tag{4}$$

as the joint conditional probability of Y_t and that the state at time *t* is $S_t = m$, given that the state at time *t-1* is $S_{t-1} = m'$.

Step 1) $\alpha_0(m)$ and $\beta_t(m)$ are initialized according to:

$$\alpha_0(0) = 1, \text{ and } \alpha_0(m) = 0, \quad \text{for } m \neq 0 \tag{5}$$

and

$$\beta_\tau(0) = 1, \text{ and } \beta_\tau(m) = 0, \quad \text{for } m \neq 0 \tag{6}$$

Step 2) As soon as Y_t is received, the decoder computes:

$$\alpha_t(m) = \sum_{m'=0}^{M-1} \Pr\{S_{t-1} = m'; S_t = m; Y_1^t\} = \sum_{m'} \alpha_{t-1}(m') \cdot \gamma_t(m',m) \tag{7}$$

and

$$\gamma_t(m',m) = \sum_X p_t(m \mid m') \cdot q_t(X \mid m',m) \cdot R(Y_t, X) \tag{8}$$

where $p_t(m \mid m')$ are the transition probabilities of the trellis, $q_t(X \mid m', m)$ the output, where X belongs to some finite discrete alphabet, and $R(Y_t, X)$ are the derived block transition probabilities.

Step 3) Once the complete sequence Y_1^t has been received, the decoder recursively computes:

$$\beta_t(m) = \sum_{m'=0}^{M-1} \Pr\{S_{t+1} = m'; Y_{t+1}^{\tau} \mid S_t = m\} = \sum_{m'} \beta_{t+1}(m') \cdot \gamma_{t+1}(m, m') \quad (9)$$

and the APP can then be calculated as:

$$\sigma_t(m', m) = \Pr\{S_{t-1} = m'; Y_1^{t-1}\} \cdot \Pr\{S_t = m; Y_t \mid S_{t-1} = m'\} \cdot \Pr\{Y_{t+1}^{\tau}, S_t = m\} = \alpha_{t-1}(m') \cdot \gamma_t(m', m) \cdot \beta_t(m) \quad (10)$$

It should be noted that the BCJR has to go through the trellis twice, once for calculating $\alpha_{t-1}(m')$ and $\gamma_t(m', m)$ while the sequence is being received and once recursively to calculate $\beta_t(m)$ and the APP.

D. Turbo Decoding

If some sort of concatenated coding scheme is used, iterative or turbo decoding can be used. The idea with iterative decoding is to divide the aposteriori probabilities, obtained from the BCJR algorithm, into three parts, a priori information, intrinsic information and most importantly extrinsic information. The extrinsic information is the indirect information you gain about a particular data bit from the other bits in the concatenated scheme. This extrinsic information is the fed back into the decoder as a priori information and the information is decoding once again, but this time with enhanced a priori knowledge, see Fig. 4.

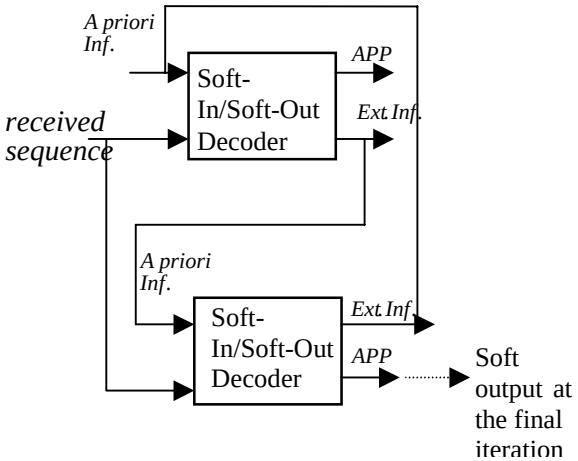

Fig. 4. Iterative (Turbo) Decoding

The advantage of turbo decoding is that one approaches the optimality of the MAP decoder with each iteration, but a much lower decoder complexity is required. The BCJR algorithm used in an iterative decoding scheme is now an option to the low complexity Viterbi algorithm.

In the DDC scheme the turbo decoding scheme may be used in conjunction with ARQ. The receiver may, for example, choose to do additional iterations on the already received results instead of asking for a new transmission, if the channel is sensitive to multi-user interference or alternately, ask for more information while iterating the already received data.

IV. Conclusions

We have put forward a framework for transmitting real-time data over a radio channel based on the Quality of Service parameters deadline (t_{DL}) and probability for correct delivery before deadline (P_d). Initial studies show a high probability of delivering the data before deadline. In this ongoing work we use a set of tools for handling information, such as hybrid ARQ, trellis representation of linear block codes, the Viterbi algorithm, the BCJR algorithm and Turbo decoding. These methods are used in conjunction with the deadline dependent coding (DDC) scheme in order to maximise the probability of delivering a message before a given deadline.

The DDC-protocol strategies for using the tools in different situations are not yet studied in detail. As an example, when high probability, P_d is required and the deadline, t_{DL} is small, two simultaneous actions can be taken: require more information from the transmitter and simultaneously performing some turbo decoding iterations for maximising the probability of correctly decoding the data block before deadline.

The required deadline and probability of delivery can be negotiated by the application, thus forming a flexible transmission mechanism. We believe that DDC might be a way of making it possible to use wireless radio channels in time- and safety critical applications.

V. References

[1] H. Bengtsson, E. Uhlemann and P.-A. Wiberg, "Protocol for wireless real-time systems", *Proc. of the 11th Euromicro Conference on Real Time Systems, York*, England, UK, June 9-11, 1999.

[2] S. Lin and D. J. Costello, Jr., "*Error Control Coding: Fundamentals and Applications*", Prentice-Hall, Inc. Englewood Cliffs, New Jersey 07632, 1983.

[3] L. R. Bahl, J. Cocke, F.Jelinek, and J. Raviv, Optimal decoding of linear codes for minimizing symbol error rate, *IEEE Transaction on Information Theory*, vol. IT-20, pp. 284-287, Mar. 1974.

[4] J. K. Wolf, "Efficient maximum-likelihood decoding of linear block codes using a trellis", *IEEE Transaction of Information Theory*, vol. 24, pp. 76-80, 1978.

[5] S. Lin, T. Kasami, T. Fujiwara and M. Fossorier, *Trellises and Trellis-Based Decoding Algorithms for Linear Block Codes*, Kluwer Academic Publishers, 1998.

[6] G. D. Forney, Jr.,"The Viterbi Algorithm", *Proceedings IEEE*, Vol. 61, pp. 268-278, 1973.