INFLUENCES OF RAKE RECEIVER/TURBO DECODER PARAMETERS ON ENERGY CONSUMPTION AND QUALITY

Lodewijk T. Smit, Gerard J.M. Smit, Paul J.M. Havinga, Johann L. Hurink[†] and Hajo Broersma[†]
Department of Computer Science,
[†]Department of Mathematical Sciences,
University of Twente, Enschede, the Netherlands
email:`smitl@cs.utwente.nl`

Abstract— **An energy-efficient architecture is vital for the next generation of mobile phones. The battery operated nature of these systems requires them to be designed and operated in a highly energy-efficient manner to maximize the battery lifetime. In this paper the characteristics (in terms of power consumption and quality) of a RAKE receiver in combination with a turbo decoder are considered. Important parameters are selected and their influences on the energy consumption and quality are investigated by means of simulations.**

I. INTRODUCTION

NEXT generation telephones (third generation and beyond) require a lot of processing power. These devices make use of a computational-intensive wideband code division multiple access (WCDMA) receiver and may use sophisticated forward error correction (FEC) methods, like turbo codes. As a consequence, next generation mobile phones consume a lot of energy. To achieve an acceptable usage time before the battery is empty, an energy-efficient architecture is a vital requirement.

Optimization for energy efficiency is not limited to application of low power hardware. Low power hardware is a first requirement to achieve an energy efficient architecture. Additionally, optimal control of this low power hardware is needed to save energy consumption.

Furthermore, an adequate quality of the wireless link is desired, which is not trivial due to the changing conditions of the external environment.

The required processing power for a WCDMA receiver, as well as the resulting quality of received frames, depends on a lot of parameters with a complex relationship between them. The challenge is to find the set of parameters that minimizes the energy consumption of the receiver, while satisfying the required quality constraints. The goal of this article is to investigate the role of three important parameters for the RAKE receiver in combination with a forward error correction turbo decoder, with regard to the energy consumption and the achieved quality. These investigations are on a functional level and are not implementation dependent.

The presented results will be used to construct a control system that is able to adapt the WCDMA receiver in order to minimize the energy consumption, while satisfying the quality constraints at run-time. These adaptations should be done at run-time due to the continuously changing external environment.

A short description of the basic principles of the RAKE receiver and the turbo decoder is given in the next section, followed by the cost and performance characteristics of these devices. To understand the effect of changing different parameters, a third generation link is simulated with a realistic channel model, including multiple users that are transmitting simultaneously, different paths, fading effects and so on. The fourth section describes the setup of this simulation environment. In the fifth section, the final results of the simulations are presented, followed by a short discussion, our conclusions and future work.

II. UMTS RECEIVER

In this paper, we consider a RAKE receiver in combination with a turbo decoder that is used in a mobile UMTS terminal. We consider only the reception of downlink traffic (transmission from base station to the mobile) because energy efficiency is more important for the mobile than for the base station.

A. Rake Receiver

A RAKE receiver (see Figure 1) is used for wideband code division multiple access systems (WCDMA) [6]. In a WCDMA system all the users transmit in the same band simultaneously. Each transmitted bit is spread by the transmitter by means of a multiplication with a pseudo random code. The length of this code is called the spreadingsfactor. A larger spreadingsfactor gives a better resistance against interference (interference of multiple users, multiple channels, multiple paths). The receiver despreads the received signal by multiplication with exactly the same PN-code. The results of all multiplications are added. This process of multiplication and addition is called correlation.

A RAKE receiver has multiple fingers to correlate the received signals from different paths with different delays, and combines the results of the different paths to construct one output signal. This basic principle of a RAKE receiver is shown in Figure 1. For a more detailed description, see [6], [9].

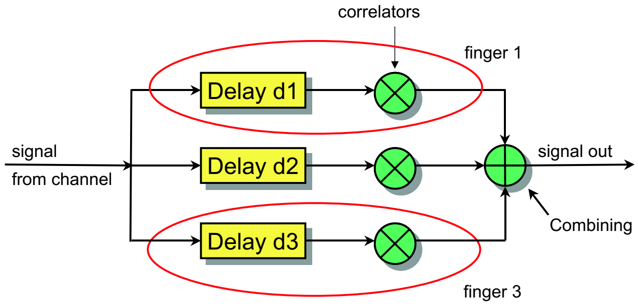

Fig. 1. Basic Principle Of A Rake Receiver

B. *Turbo Decoder*

A turbo decoder is a forward error correcting decoder, which uses the soft values of a RAKE receiver as input and produces hard bits (0 or 1) on the output. A block diagram of a turbo decoder is shown in Figure 2. A turbo decoder is constructed out of two decoders, an interleaver, and a deinterleaver. The turbo decoding algorithm can run multiple times on the same data to improve the output of a turbo decoder. This iterative principle gives a turbo decoder a better performance than a conventional decoder. More details about turbo decoding can be found in e.g. [10].

III. COST AND QUALITY CHARACTERISTICS

The main goal is to minimize the energy costs given a set of constraints, especially with regard to the required quality. In this section the costs and quality characteristics of the RAKE receiver and turbo decoder will be discussed.

There are a lot of parameters that affect the quality (and consequently the costs). However, many parameters are determined by the external environment (e.g. number of paths, number of users and interference). These ”external” parameters have a considerable influence on the system. Therefore, the effects of changes of these parameters should be taken into account, but the system cannot change them. Furthermore, the values of some parameters are determined at design time (e.g. chiprate) and cannot be changed (easily) at run-time.

A. *Cost*

The energy consumption costs are expressed in number of operations (like addition, multiplication) that are needed for the datapath. This is a rough estimation of the energy costs in relative terms. The real energy consumption in absolute terms (e.g. Joules/bit) is implementation dependent. Our goal is to make a trade-off between different set of parameters to minimize the energy consumption while satisfying an adequate quality, regardless of the exact absolute costs.

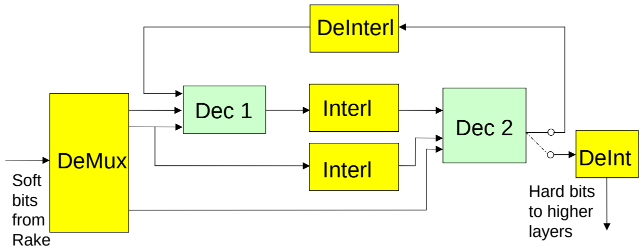

Fig. 2. Block Diagram Of Turbo Decoder

1) Rake receiver: In essence each finger of a RAKE receiver performs a correlation of the incoming chips (transmitted pulses) with a pseudo noise code (PN-code). This means that per correlator sf multiplications per bit have to be done, where sf is the spreadingsfactor. These multiplications are relative simple, because the PN-codes only contain 1 and -1 elements. Therefore, the costs for all operations (including multiplication) are considered to be the same. Additionally, the results of the different chips have to be summarized, requiring $sf - 1$ additions (in our implementation). Therefore the number of operations are about $2 \cdot sf$ per processed bit. So, the total number of operations per bit needed for the arithmetic of the RAKE receiver are about $2 \cdot sf \cdot co$, where co is the number of correlators. Note that correlators are needed for channel estimation and searching as well, so the number of correlators is larger than the number of fingers. Furthermore, most RAKE receiver designs are more complex (having tracking algorithms, advanced filters, etc.) that make the performance better and the arithmetic complexity higher. The exact costs depend on the design and implementation, but the most relevant observation is that the costs are linear proportional with the number of fingers and the spreadingsfactor. In our simulation we use the rough estimation that the costs per bit for the RAKE receiver are proportional to $2 \cdot sf \cdot co$.

2) Turbo decoder: With regard to the turbo decoder, the number of operations per bit is approximately linear with the number of iterations of the turbo-decoding algorithm. There exist different turbo decoding algorithms with different costs and performance. The two used decoders use the Log-MAP algorithm [7], which (in our situation) costs 213 operations per bit [7].

A turbo decoder contains also two (de)interleavers. The costs for these (de)interleavers are neglected as this is only addressing a memory in the right manner. Each iteration, the decoding algorithm is executed twice. The number of operations needed for the turbo decoder per bit is about $n \cdot (2 \cdot 213)$, using the Log-MAP algorithm and an encoder that is conform the UMTS specification [2], where n is the number of iterations. So, the costs are linear with the number of executed iterations. Note that these costs are only the arithmetic costs, e.g. control costs are not included.

B. *Quality*

Due to the unpredictable external environment and the complex relationship between many parameters, it is not feasible to describe the turbo/RAKE system in an analytical way. The processing costs as a result of the parameter

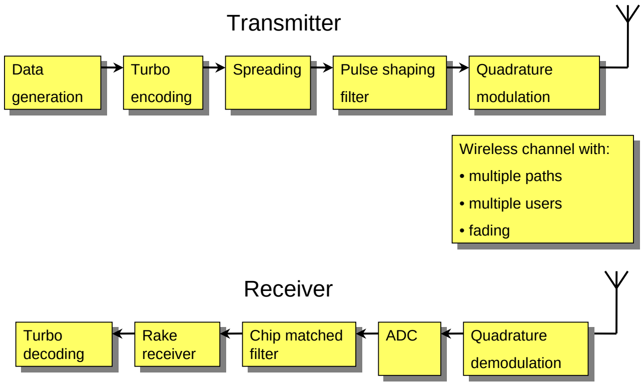

Fig. 3. 3gpp Simulation System

settings can be estimated, but the quality of the output of the turbo/RAKE system is difficult to determine in advance. To determine the quality of the output for different parameter settings, we constructed a simulation environment. With this simulation environment we can study the effects of different parameter settings. With the results of the simulations, we will try to construct heuristics about what to do in a particular case. The final goal is to construct a control system that uses real-time measurements in combination with the above mentioned heuristics to make the appropriate decisions at run-time.

The metric used for the quality is bit error rate (BER) per frame (=number of errors per frame / length of frame). The BER is determined per frame in order to study the distribution of the errors over the different frames. This distribution is relevant to know, because a frame that contains errors after the turbo decoder is useless for most applications.

IV. SIMULATION SETUP

To simulate the system, we have to simulate the receiver and FEC turbo decoder, but also the FEC turbo encoder, the transmitter and the channel. In Figure 3, an overview of the system is depicted. Below we will give a short explanation of each component. For the whole system, a lot of different parameter values have to be chosen. In most cases, we will choose the values that are suggested by the UMTS standard [1]. We can simulate a realistic wireless environment, including multiple (moving) users, multiple paths, a time-variant fading channel and power degradation.

A. *Data Generation*

The data generator generates blocks with random bit values. The block length may vary over the different simulations. Within the UMTS specification, the block length for the turbo encoder should be between 40 and 5114 bits. The simulations shown in this article are all performed with a block length of 1000 bits.

B. *Turbo Encoder*

The turbo encoder [2] encodes the information bits. According to the UMTS specification, the turbo encoder is a parallel concatenated convolutional coder with two 8-state constituent encoders and one internal interleaver. The turbo encoder has rate 1/3. The output bits of the turbo encoder are modulated; 0 and 1 are respectively mapped to -1 and 1. The bits of the second convolutional encoder are interleaved with a 3gpp [2] interleaver.

C. *Transmitter*

The transmitter spreads the incoming data and generates sf chips for every bit, where sf is the spreadingsfactor. In UMTS, the sf is between 4 and 512. The PN-code used for the spreading is generated according to the UMTS downlink scrambling code generator [3]. After the spreading, the chips are pulse shaped with a square root raised cosine filter with a roll-off factor of 0.22 and a FIR length of 17. At last, the chips are RF modulated using quadrature modulation.

D. *Wireless Channel*

The channel is modeled as a fading multipath channel with different users that are simultaneously transmitting data. The Doppler frequency is for all simulations fixed to 37 Hz. This corresponds to a velocity of the mobile of 20 km/hour. The receiver gets (in most cases) different reflections of the same transmitted signal that traverse along different paths due to obstacles in the surrounding, like large buildings, mountains, trees and so on. The different paths are modeled with delays that are multiples of the chip arrival time. In our model, the second path arrives two chip times later than the first path, the third path arrives four chip times later than the first path and so on. The reflection have equal power. All the simulations in this paper are performed with three paths. WCDMA is user interference limited [5]. Therefore it is important to model the channel with different simultaneously transmitting users. We made the assumption that the receiver receives all the different users with equal power.

E. *Rake Receiver*

At the receiver side, the received signal is RF demodulated and sampled with an oversample factor of 4. Subsequently the data is converted to digital by a digital/analog converter using 6 bits for quantization, which is sufficient according to [4]. Next, for each path that is considered, a finger of the RAKE receiver, see Figure 1, does the correlation. We assumed perfect channel estimation and tracking, so that each path is known to the RAKE receiver and there is perfect synchronization.

F. *Turbo Decoder*

The turbo decoder consists of two 8-state Soft Input Soft Output (SISO) decoders, separated by an interleaver that is specified by the UMTS specification [2]. The SISO decoders make use of the LOG MAP algorithm [7].

V. SIMULATION RESULTS

A. *Turbo Decoder Decoding Limit*

The goal of our first simulation is to investigate the relation between the input of the turbo decoder (=output of the RAKE receiver) and the output of the turbo decoder. Our objective is to find for a frame the maximum amount of errors that the turbo decoder is able to correct. In Figure 4 we show the BER from the RAKE output versus the BER from the turbo decoder output after 10 iterations, under poor channel conditions. Poor channel conditions are chosen, because the energy consumption of the receiver to receive bits from a bad channel is higher than for a good channel. Each cross represent both values for a specific frame.

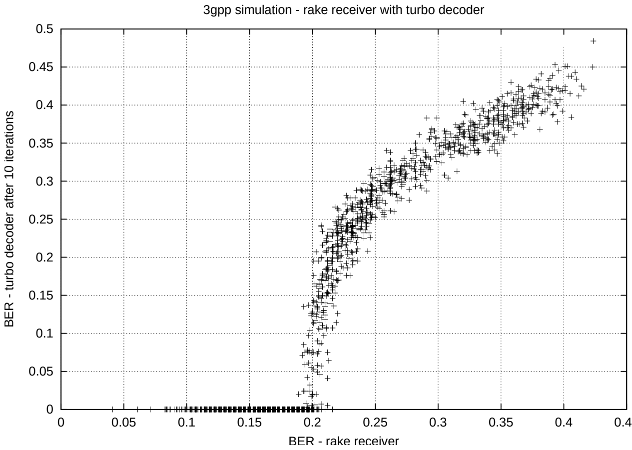

Fig. 4. Limits of turbo decoding

With this plot, a prediction can be made about the prospect that the turbo decoder can correct the frame as a function of the number of errors in the frame that comes from the RAKE receiver. As can be seen from the plot, this prediction can be made very accurately. If the BER of the RAKE output is more than 0.2, turbo decoding can not recover all the errors in the frame. Applying turbo decoding on such frames is useless and a waste of energy. If the BER at the RAKE output is smaller than 0.18, the turbo decoder is almost always able to recover all the errors in the frame. The BER range of the RAKE output in which there is a high uncertainty about what the result will be after the turbo decoder is small.

Note that these results are for a channel with a random error distribution. Within UMTS, there is a large interleaver before the turbo encoder, so the error distribution assumption is valid most of the time.

B. *Rake Receiver Performance*

Two important parameters of the RAKE receiver with regard to energy consumption and the quality of the output are the number of fingers and the spreadingsfactor. Another important parameter that has a considerable influence on the quality is the number of users. These parameters are varied in our simulation to investigate the effect on the quality; the other parameters are conform Section IV, with SNR=1 dB. All the possible combinations with the following parameters have been simulated: number of users = {6,24}, spreadingsfactor = {8,16,32,64} and number of fingers = {1,2,3}. The ranges have been limited to create an understandable plot. For each frame, the number of errors in the received frame is counted. This is converted to a BER for each individual frame. The results are plotted in Figure 5. For each set of parameters, one cross is plotted with the spreadingsfactor printed near the cross. The costs of each simulation result are computed according to the formula in Section III and the quality is the maximum observed BER of all the received frames. We are interested in the maximum observed BER, because we should be able to turbo decode even the frame with the worst quality. The figure shows lines for equal number of fingers and equal number of users with different spreadingsfactors. The three lines in the left bottom corner are the situations with 6 users. The other lines are the situations with 24 users.

As shown in Section V-A, the quality of the RAKE output should have a BER less than 0.18. Otherwise, the turbo decoder is not able to correct the frames. Furthermore, the costs should be as low as possible. For the situation with 24 users, two sets of parameters with equal costs qualify for these requirements: {sf=64, fingers=1} or {sf=32, fingers=2}. The first set has a better quality (lower BER). The second set has a lower spreadingsfactor, resulting in a double bandwidth with the same chiprate.

With this kind of plots, it is easy to identify the most suitable set of parameters. Depending on the requirements of the given application(s) the most suitable set of parameters can be chosen.

C. *Rake Receiver - Distribution of Errors*

We showed that when we know the number of errors in the RAKE output, we can make a good trade-off between different sets of parameters. However, in reality, we do not know the exact number of errors in a frame because the transmitted frame is unknown. Therefore, it is relevant to have an adequate prediction of the number of errors in a frame. In Figure 6, the error distribution is shown for one specific parameter set (24 users, spreadingsfactor=32, 3 fingers). The number of errors per frame is converted to a BER.

The distribution of the errors is relative narrow. Other sets of parameters (not shown here) show a similar distribution. If we can detect the peak of the graph and we know the width of the graph, we can compute the right side of the graph.

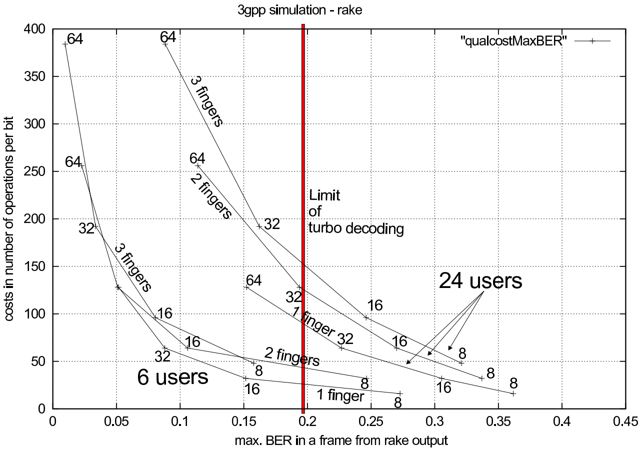

Fig. 5. Different parameter settings

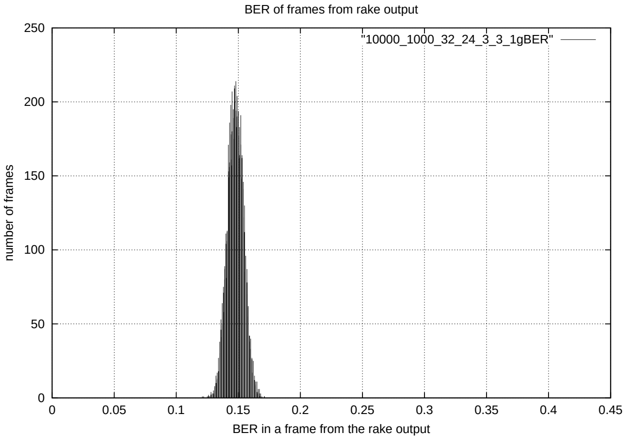

Fig. 6. RAKE receiver - error distribution

This is relevant, because the right side of the graph should be below 0.18 (as we concluded before).

To detect the peak, we try to estimate the average number of errors per frame through observation of the soft output of the RAKE receiver. We apply a simple algorithm on the soft output of the RAKE receiver to perform the estimation. If the absolute output of the RAKE receiver for one specific bit is above a defined threshold, the bit is assumed to be received correctly. Otherwise, the bit is assumed to be received incorrectly. With this simple approach, an accurate estimate of the current quality can be obtained. The estimation deviation is within 5% of the real number of errors. The estimation is not perfect, but good enough for our purpose. Note that an exact approximation is not at all possible due to the changing conditions.

D. *Turbo Decoder Performance*

In Table I, the performance of the turbo decoder is shown for the parameter set {users=24, sf=32, fingers=3}. The number of frames with errors is shown, as a function of the number of turbo decoder iterations. The range of different number of the turbo decoder iterations is very small. In almost all cases at least two and at most three turbo decoding iterations are needed. Therefore, for one specific situation, the profit of changing the number of turbo decoding iterations on the fly is limited. If the quality output of the RAKE receiver is much better, the turbo decoder needs only one iteration. Therefore, it is useful to determine the number of turbo decoder operations for different sets of parameters. The quality estimation of the RAKE output can be used for a first guess of the number of needed turbo decoding iterations. Further, there are early stop algorithms known that try to determine when to stop the turbo decoder, see e.g. [8], [11].

VI. DISCUSSION

The results presented in the previous section can be used to build a control system that adapts the receiver with an optimal set of parameters, based on real-time measurements. With our approach, the quality of the output of the RAKE receiver is as low as possible, and the limit about what the turbo decoding is able to correct is approached as closely as possible. This approach is quite different from the conventional view on turbo coding. Turbo coding is commonly used to make a signal with a perfect quality from a good signal. We use the turbo codes to achieve a good signal from a bad signal. Our quality output is not perfect, which is acceptable for many applications. For example, in a video application a frame may occasionally be skipped, or higher protocol layers may decide to retransmit a frame that has not been decoded correctly, if extra latency is allowed. In exchange of these limited amount of frames with errors, attractive savings on the energy consumption may be reached. The last improvement in quality requires a lot of effort, so if a few frames with errors can be tolerated, the receiver can be much more energy-efficient.

VII. CONCLUSIONS

In this paper, we investigated on a functional level a RAKE receiver in combination with a turbo decoder from an energy efficiency viewpoint.

no. of td it.	# bad frames	percentage
after RAKE	10000	100 %
after 1 td it.	9279	93 %
after 2 td it.	71	0 %
after 3 td it.	1	0 %
after 4 td it.	1	0 %
after 5 td it.	0	0 %

TABLE I

NUMBER OF FAULT FRAMES AS FUNCTION OF NR OF TD ITERATIONS

We demonstrated with simulations that the turbo decoding algorithm has a very well defined working area, defined in terms of the BER of the incoming frame, provided that the errors are randomly distributed over the frame. Also we demonstrated that for a specific set of parameters, the number of iterations of the turbo decoder to correct a frame is almost always the same.

A simple algorithm is sketched that can effectively predict at real-time the number of errors in a frame that is supplied to the turbo decoder.

With simulations for multiple sets of parameters it is shown that choosing the optimal parameter set can have a considerable impact on the energy consumption of the mobile. Choosing a suitable parameter set is possible through the error prediction algorithm combined with the knowledge about the characteristics of the turbo decoder.

VIII. FUTURE WORK

Based on the presented results, a control system will be built that can minimize at run-time the energy consumption while satisfying the quality of service constraints.

More investigations in parameters like the blocksize and the puncturing rate are required to achieve a more complete model. Increasing the blocksize will increase the performance of the turbo decoder without an increase in energy consumption, but the latency will increase. Puncturing will degrade the performance of the turbo decoder and also decrease the energy consumption.

Using a lower spreadingsfactor will result in a higher bandwidth. In this case the receiver (including the analog part) can be switched off sooner, saving energy. The analog part of the receiver should be included in the model because this part is responsible for a considerable part of the energy consumption of a receiver.

The cost of the effectuation of decisions should be included in the model. For example, for a change in the spreadingsfactor a negotiation with the base station is required.

Additional quality constraints like a minimal throughput should also be taken into account.

Acknowledgements

This research is conducted within the Chameleon project (TES.5004) supported by the PROGram for Research on Embedded Systems & Software (PROGRESS) of the Dutch organization for Scientific Research NWO, the Dutch Ministry of Economic Affairs and the technology foundation STW. We would like to thank the people from the Turbo Code Research team at the University of Virginia (UVa) for providing the matlab sources for turbo code simulations. Also we would like to thank the people from the institute of microelectronic systems from the Technical University of Darmstadt (Germany) for providing the matlab sources for WCDMA simulations. We would like to thank Jos A. Huisken, Kees G.W. Goossens and John T.M.H. Dielissen from Philips Research in Eindhoven for their contributions to this work.

REFERENCES

[1] http://www.3gpp.org.

[2] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network. *3G TS 25.212 v4.1.0 (2001-06): Multiplexing and channel coding (FDD) (Release 4)*, June 2001. see: http://www.3gpp.org.

[3] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network. *3G TS 25.213 v4.1.0 (2001-06) : Spreading and modulation (FDD) (Release 4)*, June 2001. see: http://www.3gpp.org.

[4] J. Becker, T. Pionteck, and M. Glesner. Simulation, prototyping and reconfigurable hardware realization of cdma rake-receiver algorithms for flexible mobile transceivers. In *Proc. of ERSA'01*, jun 2001.

[5] T. F. M. Bossert. Interference cancellation in the synchronous downlink of cdma-systems. In *ITG-Fachtaging : Mobile Kommunication*, pages 331–338, Sept. 1995.

[6] T. Ojanperä. *Wideband CDMA for Third Generation Mobile Communications*. The Artech House Universal Personal Communications Series. Artech House, 1998. ISBN: 0-89006-735-X.

[7] P. Robertson, E. Villebrun, and P. Hoeher. A comparison of optimal and sub-optimal map decoding algorithms operating in the log domain. In *Proc. International Conference on Communications (ICC)*, pages 1009–1013, June 1995.

[8] R. Y. Shao, S. Lin, and M. P. Fossorier. Two simple stopping criteria for turbo decoding. *IEEE Transactions on Communications*, 47(8):1117–1120, Aug. 1999.

[9] G. L. Turin. Introduction to spread-spectrum antimultipath techniques and their application to urban digital radio. *Proc. of the IEEE*, 68(3):328–353, Mar. 1980.

[10] M. C. Valenti. *Iterative Detection and Decoding for Wireless Communications*. PhD thesis, Virginia Polytechnic Institute and State University, July 1999.

[11] A. Worm, H. Michel, F. Gilbert, G. Kreiselmaier, M. Thul, and N. Wehn. Advanced implementation issues of turbo-decoders. In *Proc. 2nd International Symposium on Turbo-Codes and Related Topics*, Sept. 2000.