ESIT 2018 Tutorials

Incremental Redundancy

Richard Wesel
University of California, Los Angeles

Abstract

This tutorial explores approaching capacity by using incremental redundancy (i.e. a variable-length code) as an alternative to using a code with a fixed (and relatively long) codeword length. Systems that use incremental redundancy can adaptively decrease the attempted information rate until it falls below the accumulated information density per symbol or “operational capacity” of the channel. This dynamic adaptation can approach capacity with shorter (average) codeword lengths and/or lower complexity than systems that must rely on long codeword lengths (and ergodicity) to allow the accumulated information density per symbol to concentrate around the mutual information of the channel.

Shannon showed that even complete and noiseless feedback of everything the receiver knows does not improve the capacity of a memoryless channel. Feedback does, however, improve the error exponent, which suggests that it should facilitate shorter blocklengths. Indeed, several approaches show that fully utilizing noiseless feedback of all received symbols allows capacity to be approached with very short average codeword lengths, and this tutorial will briefly describe one such technique: Active Sequential Hypothesis Testing (ASHT).

The cost of noiseless (or extremely reliable) feedback of everything the receiver knows can outweigh its benefit, and many practical systems use feedback primarily to identify when the transmission of incremental redundancy should be terminated (i.e. by sending and ACK or NACK). Polyanskiy et al. have quantified the benefit of this so-called “stop feedback” through an enlightening finite-blocklength analysis, providing random coding lower bounds on the throughputs possible with short average codeword lengths when feedback is used only to facilitate termination.

At the heart of practical systems using incremental redundancy is a variable-length code (also called a rate-compatible code) whose encoding and decoding structure allow codeword length to grow as additional incremental redundancy is made available. This tutorial provides an overview of common techniques for rate-compatible code design using convolutional codes, binary low-density parity check (LDPC) codes and their non-binary counterparts. A central insight to the development of effective rate-compatible codes is to view the highest rate code as the “actual” code and the subsequent

incremental redundancy as additional information that is used only to support the decoding of the highest rate code.

In practical systems where feedback is used to terminate transmission, the receiver must ascertain when it has decoded with sufficient reliability without an undue overhead on codeword length. This tutorial presents a recent technique that shortens cyclic redundancy check (CRC) length by incorporating the structure of the inner code into the CRC design. Raghavan and Baum's technique for enhancing a Viterbi decoder to compute the probability that its decoding decision is in error can avoid the need for a CRC altogether, at the cost of additional receiver complexity. The tutorial presents a lower-complexity implementation of the Raghavan and Baum technique.

Regardless of the receiver strategy for declaring termination, practical systems, must identify a small number of decoding points in the sequence of incremental redundancy (i.e. the lengths of the incremental redundancy packets that will be sent). The tutorial presents the sequential differential optimization technique for determine these packet lengths to maximize throughput.

Packet-level erasure codes (e.g. Fountain codes) allows a transmitter to serve multiple receivers simultaneously while requiring each receiver to decode only the minimum number of packets that it needs. This tutorial presents a simple derivation of the key equation governing peeling decoder performance of such erasure codes.

Finally, the tutorial explores feedback-free incremental redundancy techniques, which allow a large number of capacity-approaching variable-length codes with short average codeword length to approach capacity by sharing a common pool of redundancy that can be used by exactly the decoders that need without requiring feedback.