A Comparison Between Two Error Detection Techniques Using Arithmetic Coding

Bin He and Constantine N. Manikopoulos
Department of ECE New Jersey Institute of Technology Newark, NJ 07102
Email: b.xh0843@oak.njit.edu manikopoulos@adm.njit.edu

This paper compares the error detection capability of two joint source channel coding schemes with arithmetic coding, the forbidden space approach [1] and marker symbol approach [2].

For forbidden space approach, it is modeled as geometric distribution with random variable $Y$ [3]. The probability that error propagates more than $n$ symbols is $P_1[Y > n] = (1 - \epsilon)^n$, where $\epsilon (0 < \epsilon < 1)$ is size of forbidden space. The redundancy is $R_1 = -\log_2(1 - \epsilon)$ bits per symbol.

For the marker symbol approach, the redundancy is $R_2 = \frac{c}{m}$ bits per symbol, where one symbol contains $c$ bits. The misdetection probability of one marker is found to be $2^{-c}$. For the same amount of redundancy, i.e., $R_1 = R_2$, we have $(1 - \epsilon)^m = 2^{-c}$. This means two approach basically have the same error detection capability. The misdetection probabilities in each marker block size are the same for both approaches.

While two approaches basically have the same detection capability, the forbidden space approach is more efficient in error correction because we can estimate error location based on the geometric distribution. On the other hand, the marker symbol approach is simpler and ready to use in packet switching, where the forbidden space approach with continuous error checking does not fit.

In our simulation, we find error introduced in arithmetic decoding process may generate a long run of correctly decoded symbols between two incorrect symbols. The reason is with error introduced and an incorrect symbol decoded, the decoding interval may still be correct. The next incorrect symbol cannot be decoded until we come to use the incorrect part in the frequency table caused by the error and the impact of the incorrectly decoded symbol to the frequency table accumulates big enough. Thus a long run of correct symbols are generated between two incorrect symbols.

References

