PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL

FACULDADE DE ENGENHARIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA - PPGEE

DIEGO VIER LOSS

DECODIFICADOR CONVOLUCIONAL DE VITERBI COM *SOFT-DECISION* E BIT *INTERLEAVER*

DIEGO VIER LOSS

Porto Alegre, março de 2008.

DECODIFICADOR CONVOLUCIONAL DE VITERBI COM *SOFT-DECISION* E BIT *INTERLEAVER*

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Faculdade de Engenharia da Pontifícia Universidade Católica do Rio Grande do Sul, como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica.

Área de concentração: Tecnologia da Informação / Telecomunicações.

Orientador: Prof. Dr. Fernando César Comparsi de Castro.

Porto Alegre, março de 2008

"DECODIFICADOR CONVOLUCIONAL DE VITERBI COM *SOFT DECISION* E *BIT INTERLEAVER* "

DIEGO VIER LOSS

Esta dissertação foi julgada para a obtenção do título de MESTRE EM ENGENHARIA e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Pontifícia Universidade Católica do Rio Grande do Sul.

Fernando César Comparsi de Castro, Dr.
Orientador

Rubem Dutra Ribeiro Fagundes, Dr.
Coordenador
Programa de Pós-Graduação em Engenharia Elétrica

Banca Examinadora:

Fernando César Comparsi de Castro, Dr.
Presidente – PUCRS

Dênis Fernandes, Dr.
PUCRS

Maria Cristina Felippetto de Castro, Dra.
PUCRS

Dedico este trabalho à minha companheira de todas as horas Marcia e ao meu filho Guilherme.

Agradecimentos

Agradeço aos meus pais pelo contínuo e intenso estímulo ao investimento no campo intelectual e pelos valores por eles a mim passados, que alavancaram a minha decisão de fazer este curso. Agradeço à minha noiva Marcia, pela força que me passou e pela compreensão que teve nos momentos em que tivemos de abrir mão da companhia um do outro. E finalmente, agradeço aos meus orientadores, Fernando e Cristina e aos meus caros colegas do CPTW, pela recepção calorosa, pela paciência, pela amizade e pela carga descomunal de novas informações, idéias e conhecimentos que despejaram sobre mim.

Diego

Resumo

DIEGO VIER LOSS - **"Decodificador Convolucional de Viterbi com *Soft-Decision* e *Bit Interleaver*".**

Esta dissertação versa sobre a análise, o projeto e a implementação de um decodificador convolucional de Viterbi com *Soft-Decision* precedido por bit *interleaving*, visando a aplicação em sistemas de comunicação digital com modulação *M*-QAM. O decodificador implementado é parte do processo de desenvolvimento do sub-sistema de "Modulação inovadora" do Sistema Brasileiro de Televisão Digital (SBTVD), instituído pelo decreto Nº4.901, de 27 de novembro de 2003. O subsistema "Modulação inovadora" foi desenvolvido no Centro de Pesquisa em Tecnologias *Wireless* (CPTW) da Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS).

Tendo a PUCRS recebido do governo brasileiro a incumbência do desenvolvimento do subsistema de "Modulação inovadora", a equipe de P&D do CPTW decidiu que o codificador de canal do sistema em desenvolvimento deveria incluir *Soft-Decision* em conjunto com bit-*interleaving*, para efeito de maximizar o desempenho operacional do sistema. Esta é, portanto, a razão da escolha do tema desta dissertação.

É interessante aqui observar a dificuldade de se encontrar, para não dizer a inexistência de literatura (tanto em livros quanto em publicações acadêmicas) indicando diretivas efetivas que auxiliem na implementação prática deste decodificador. Esta implementação só foi possível a partir de inferências sobre textos acadêmicos publicados em veículos conceituados como o IEEE (*The Institute of Electrical and Electronics Engineers*), e a partir de experimentação prática e de testes de hipóteses em laboratório, trabalho este executado durante as atividades de pesquisa no Laboratório de Sistemas *Wireless* (LSW) do CPTW, vinculado a este mestrado.

Os resultados obtidos a partir do desenvolvimento destes estudos foram, também, de vital importância para o rápido e bem sucedido desenvolvimento do projeto do codificador e decodificador de canal do sistema ISDB-T. O sistema ISDB-T, japonês, foi o sistema escolhido pelo governo brasileiro para ser o sistema de televisão digital aberta adotado no Brasil. O CPTW da PUCRS recebeu do governo a incumbência de desenvolver o projeto "Desenvolvimento de Moduladores para *Broadcast* de Sinais de TV Digital" através do contrato "Concessão de Colaboração Financeira Não-Reembolsável No.06.2.1194.1.4" com o Banco Nacional de Desenvolvimento Econômico e Social (BNDES), projeto este que prevê a concepção e execução de um modulador ISDB-T em lógica programável, para posterior conversão em um chip dedicado. A construção deste modulador implica necessariamente na construção de um demodulador, para efeito de validação do projeto, tendo sido, desta forma, utilizados todos os conceitos, conhecimentos e resultados obtidos a partir do trabalho de pesquisa que originou esta dissertação.

Abstract

DIEGO VIER LOSS - **"Viterbi Convolutional Decoder With *Soft-Decision and Bit Interleaver*".**

In this work we design and implement a Viterbi convolutional decoder with Soft-Decision and bit interleaving for digital communication systems based upon quadrature amplitude modulation. This decoder is part of researches of a *Sistema Brasileiro de Televisão Digital* subsystem, called *"Modulação Inovadora"* developed integrally in the *Centro de Pesquisa em Tecnologias Wireless (CPTW)* of *Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)*. The *Sistema Brasileiro de Televisão Digital* was established by the Nº4.901 Brazilian government decree of November 27th of 2003.

Since PUCRS got the *"Modulação Inovadora"* project incumbency, the CPTW's R&D crew decided to use just cutting edge techniques to compose the most advanced and reliable modulator as possible. In this way was decided to use Soft-Decision as the mode of operation of the Viterbi decoding and that's the reason why this work treat about this theme.

It is important to say that is very difficult or almost impossible find references (books about wireless technologies or scientific and technical papers) that show or indicate the way for the practical implementation of a Viterbi decoder with these specific high performance characteristics. This implementation just was possible by the inferences made over academic papers published by world respected publications as the IEEE (The Institute of Electrical and Electronics Engineers), hypothesis tests in laboratory and practical experiments. All that work was made inside the *Laboratório de Sistemas Wireless (LSW)* of CPTW during the master degree entailed to the *"Modulação Inovadora"* project.

The research results were very important for the fast and successful subsequent developments in the CPTW, as the ISDB-T modulator and demodulator. The ISDB-T system is the Japanese broadcast digital television system that Brazil adopted as the Brazilian terrestrial digital television broadcast standard. The CPTW got the "*Desenvolvimento de Moduladores para* Broadcast *de Sinais de TV Digital*" project development incumbency with the contract "*Concessão de Colaboração Financeira Não-Reembolsável No.06.2.1194.1.4*" signed with the *Banco Nacional de Desenvolvimento Econômico e Social (BNDES).* In this contract the CPTW assumed the compromise of developing one ISDB-T modulator over an field programmable gate array (FPGA) platform to be later converted to an dedicated integrated circuit (ASIC). To do so all the concepts and knowledge obtained in these work was decisive important to develop the ISDB-T modulator and one validation test ISDB-T demodulator.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

SNR	*Signal-to-Noise Ratio* (Relação entre sinal e ruído)
ISI	Interferência Inter-Simbólica
TV	Televisão aberta
RF	Rádio Freqüência
AD	Conversor Analógico-Digital
CPTW	Centro de Pesquisa em Tecnologias *Wirwless* (PUCRS)
CODEC	Conjunto de codificador e decodificador
BER	*Bit-Error-Rate* (número de erros por bit)
SBTVD	Sistema Brasileiro de Televisão Digital
QAM	*Quadrature Amplitude Modulation*

1 Introdução

É crescente e significativa a importância das tecnologias *wireless* na sociedade em que vivemos. A forma como estamos nos organizando faz com que este modo de transmitir e receber informações ocupe um lugar cada vez mais importante em nossas vidas. A quantidade de informação que estamos transmitindo, neste contexto, vem aumentando dia após dia, levando as tecnologias envolvidas aos seus limites de desenvolvimento e aprimoramento.

Dentre os inúmeros desafios encontrados para fazer com que dados contendo informações saiam de um ponto qualquer da terra e cheguem a outro, sempre estará envolvido o esforço de manter a integridade destes. Este esforço deve-se ao fato de mesmo as tecnologias mais avançadas sempre estarem sujeitas a uma enorme variedade de possíveis fontes de interferência.

Os sistemas de telecomunicações baseados em tecnologia digital dominam o mercado de transmissão de dados devido ao seu baixo custo (se comparado aos sistemas analógicos) e grande capacidade de transporte de dados com uma utilização otimizada da banda espectral. Os códigos corretores de erros são essenciais para que sistemas deste tipo possam ter maiores alcance e confiabilidade, sendo métodos eficientes e eficazes de "corrigir" os erros inseridos pelo canal. Dentre os mais variados canais existentes para a propagação de sinais digitais está o canal *wireless*, que, embora seja muito atrativo pelo fato de que quase tudo sobre a Terra estar imerso no espaço que o define, também é um dos meios mais "hostis", com uma imensa gama de possíveis fontes de interferência e degradação dos sinais de radiofreqüência.

É a partir da comunicação de dados por radiofreqüência propagada no espaço que funciona um dos mais populares meios de comunicação de massa do mundo, e particularmente do Brasil, a televisão. Devido à grande importância deste veículo, o governo brasileiro decidiu digitalizar a transmissão do sinal de TV aberta (terrestre) para permitir que haja uma melhor utilização do espectro de RF e possibilitar a transmissão de

volumes maiores de dados. A partir desta decisão, inúmeros projetos vinculados à pesquisa sobre sistemas de transmissão e recepção de sinais digitais começaram a ser financiados com recursos públicos, de forma a viabilizar o ingresso do país nesta nova tecnologia. A Pontifícia Universidade Católica do Rio Grande do Sul, por meio do CPTW, recebeu a incumbência de desenvolver um subsistema inovador de modulação digital (módulo de "Modulação inovadora") para integrar ao Sistema Brasileiro de Televisão Digital (SBTVD), instituído pelo decreto No.4901.

Posteriormente, o governo brasileiro optou pela escolha do sistema japonês de modulação para TV digital, o ISDB-T, e a PUCRS, novamente através do CPTW, recebeu a incumbência de desenvolver um modulador conforme este padrão, dando início ao projeto, que encontra-se em andamento, "Desenvolvimento de Moduladores para *Broadcast* de Sinais de TV Digital", através do contrato "Concessão de Colaboração Financeira Não-Reembolsável No.06.2.1194.1.4" com o Banco Nacional de Desenvolvimento do Econômico e Social (BNDES).

A correção de erros através da codificação e decodificação convolucional foi utilizada em ambos os projetos, tendo sido os estudos realizados para o projeto do modulador do SBTVD de vital importância para a rápida, segura e correta aplicação dos conceitos necessários ao desenvolvimento do modulador ISDB-T. Este intercâmbio de informações, de forma tão direta, foi possível graças ao alto grau de semelhança das etapas de codificação convolucional de ambos os sistemas, que utilizam a decodificação de Viterbi com *Soft-Decision* em conjunto com *interleavers* de bits e operam com modulações M-QAM.

Este trabalho, portanto, apresenta o resultado das pesquisas e estudos realizados pelo autor no âmbito dos projetos de P&D do CPTW (Centro de Pesquisa em Tecnologias *Wireless*) da PUCRS. Os referidos estudos tratam sobre correção de erros através da codificação e da decodificação convolucional de dados transmitidos por meio de radiodifusão *broadcast* (termo usado para designar meios de transmissão como rádio AM/FM e televisão, em que um gerador de conteúdo transmite um sinal e tantos quantos forem os receptores no seu raio de alcance podem receber este sinal). Mais precisamente os estudos destinaram-se à implementação de um codificador convolucional e de um decodificador de Viterbi em linguagem ANSI-C, para posterior integração com os demais módulos dos sistemas de transmissão de sinal de televisão digital terrestre (caso em que o transmissor localizado em terra transmite por meio do meio atmosférico as informações)

sob desenvolvimento no CPTW. A implementação em ANSI-C é passo essencial para que se possam validar os conceitos pesquisados e idealizados durante a fase de planejamento e concepção dos sistemas a serem desenvolvidos. Apenas após a validação é possível, com o mínimo de esforço e máximo aproveitamento de recursos, implementar em *hardware* os projetos concebidos, tornando esta uma tarefa quase tão simples quanto a tradução de um código entre duas diferentes linguagens de programação.

Uma característica muito importante das transmissões *broadcast* é o virtualmente infinito número de possíveis receptores ligados a um único transmissor, impossibilitando que, mesmo que houvesse um canal de retorno, um dado perdido por um dos receptores pudesse ser reenviado a ele. No caso da televisão digital terrestre, por exemplo, a gama de potenciais geradores de interferência e degradação dos sinais transmitidos é maior do que na maioria dos outros meios de propagação. Uma vez que o sinal é irradiado através do meio atmosférico, ele se propaga em todas as direções e está sujeito a encontrar barreiras, pontos de reflexão estáticos e móveis, fontes de interferência contínuas e impulsivas, além de ter sua potência atenuada tanto quanto mais distantes os receptores estiverem do transmissor, tornando-o bastante vulnerável a interferências.

Esta característica faz com que a utilização de mecanismos que possibilitem a correção de eventuais erros inseridos no processo de envio dos dados da fonte ao destino seja de vital importância para a correta recuperação dos dados enviados. Há inúmeros processos aplicáveis em *hardware* ou *software* capazes de corrigir erros em dados corrompidos. Este trabalho versa sobre o estudo desenvolvido no contexto do processo de codificação convolucional e decodificação utilizando o decodificador de Viterbi com *Soft-Decision* em sistemas de comunicação digital com modulação M-QAM e com bit *interleaver* .

Há várias maneiras de se decodificar um código convolucional. A maneira que abordaremos neste trabalho será o decodificador de Viterbi, sendo citada na literatura como a ideal para sistemas de telecomunicações devido a suas eficiência e “leveza” computacional. O *Soft-Decision* é uma das variações possíveis na implementação do decodificador de Viterbi, sendo este método uma alternativa ao *Hard-Decision*. O uso do método *Soft-Decision* permite que o decodificador de Viterbi possa ter até praticamente a mesma capacidade de correção de erros que o uso do método *Hard-Decision* teria, em relação aos mesmos dados e operando sob um canal cuja SNR (relação sinal ruído) fosse,

em média, 3dB maior. Em se tratando de comunicação digital de sinais, estes 3dB são considerados um ganho significativo em termos de desempenho. Este ganho torna-se mais evidente quando ponderado juntamente com o aumento mínimo de complexidade e exigência de hardware da implementação do *Soft-Decision* em comparação com a do *Hard-Decision*. Uma vez que o que se procurava no CPTW era o desenvolvimento de um sistema de comunicação digital de alto desempenho, cada etapa do processo teria de ser implementada da forma mais eficiente possível. Isto significa que os 3dB de diferença entre o *Hard* e o *Soft-Decision* fizeram do segundo a escolha condizente com as exigências do projeto.

É no momento de implementar na prática um decodificador de Viterbi com *Soft-Decision* que começam os problemas. Ao buscar-se na literatura clássica usada como suporte para estudos na área de comunicação digital [1][2][3][4] verifica-se que a maior parte da informação está destinada ao estudo do *Hard-Decision* (modo comumente analisado na literatura especializada). Apenas nos foi possível encontrar material consistente sobre o *Soft-Decision* após uma extensa pesquisa na base de dados do IEEE, porém mesmo assim estas referências são incompletas e repletas de lacunas [5][6][7][8][9][10][11][12][13][14][15][16][17].

A implementação do *Soft-Decision* exige cuidados que devem ser tomados previamente, ainda na fase de concepção do sistema, cuidados estes parcialmente citados em apenas um dos artigos encontrados no IEEE [5]. Entre estes cuidados, citados no referido artigo ou inferidos no desenvolvimento do projeto, estão como mais relevantes a necessidade de projetar-se o modulador e o demodulador de maneira apropriada para garantir uma ordenação dos bits dos símbolos coerente ao processo de decodificação e a necessidade da saída do demodulador ser representada em *soft*-bits, conceito que será explicado adiante.

Na maior parte dos casos, e no caso do projeto de "Modulação inovadora" desenvolvido no CPTW, é utilizado, associado com a codificação convolucional, um bit *interleaver*. O objetivo do bit *interleaving*, que é feito no transmissor e desfeito no receptor, é espalhar e separar ao longo do tempo todos os bits originalmente próximos e adjacentes entre si. Esta separação possibilita que, em caso de influência de um *burst* de ruído impulsivo no canal, os erros gerados durante o intervalo de tempo deste *burst* sejam espalhados ao longo da seqüência dos dados depois de desfeito o bit *interleaving,* no

receptor. Caso os bits corrompidos pelo ruído não fossem distribuídos, a concentração destes erros em um intervalo contínuo de tempo causaria uma propagação do erro no decodificador de Viterbi, podendo faze-lo inserir mais erros, ao invés de corrigi-los [1].

O maior problema causado pelo bit *interleaver* no decodificador de Viterbi com *Soft-Decision*, é o fato de os bits que representam um dos estados do código convolucional, gerado no transmissor, não serem transmitidos em um mesmo símbolo da modulação utilizada. Como conseqüência, não é possível determinar o erro, na recepção, contido naquele grupo de bits, oriundos de símbolos independentemente transmitidos em instantes de tempo distintos. Para solucionar o problema causado pelo bit *interleaving,* usa-se o conceito de *soft*-bits [5], já mencionado anteriormente. Neste método, atribuí-se a cada bit, contido em um símbolo transmitido, um "grau de colaboração" para a posição deste símbolo na constelação. Desta forma é possível fazer com que o bit carregue consigo um valor que represente o erro de posicionamento do símbolo recebido em relação aos símbolos da constelação de referência. Para tanto é preciso representar um bit usando-se uma variável que permita variar seu valor em mais níveis de quantização do que simplesmente um ou zero, possibilitando indicar o quão mais próximo de zero ou de um o bit se encontra, em conseqüência do erro de posição do símbolo que o representa. Computacionalmente, o uso dos *soft*-bits em lugar de simplesmente bits, significa apenas a necessidade de um aumento de memória para o armazenamento da representação "*soft"* de cada bit de dado.

2 Princípios de *broadcast wireless* digital

Em um sistema de transmissão de televisão digital há vários meios possíveis de serem usados para o transporte do sinal. Neste trabalho trataremos de televisão digital aberta, ou também chamada de terrestre. Este tipo de sistema usa o ar como meio de propagação do sinal de rádio freqüência. É possível sintonizar e receber este sinal em toda a área de abrangência alcançada pelo transmissor. Esta topologia de rede de transmissão e recepção é conhecida como *broadcast* (um transmissor e vários receptores, usado para transmissão de televisão aberta (TV), rádios AM e FM, conforme ilustrado na Figura 2.1).

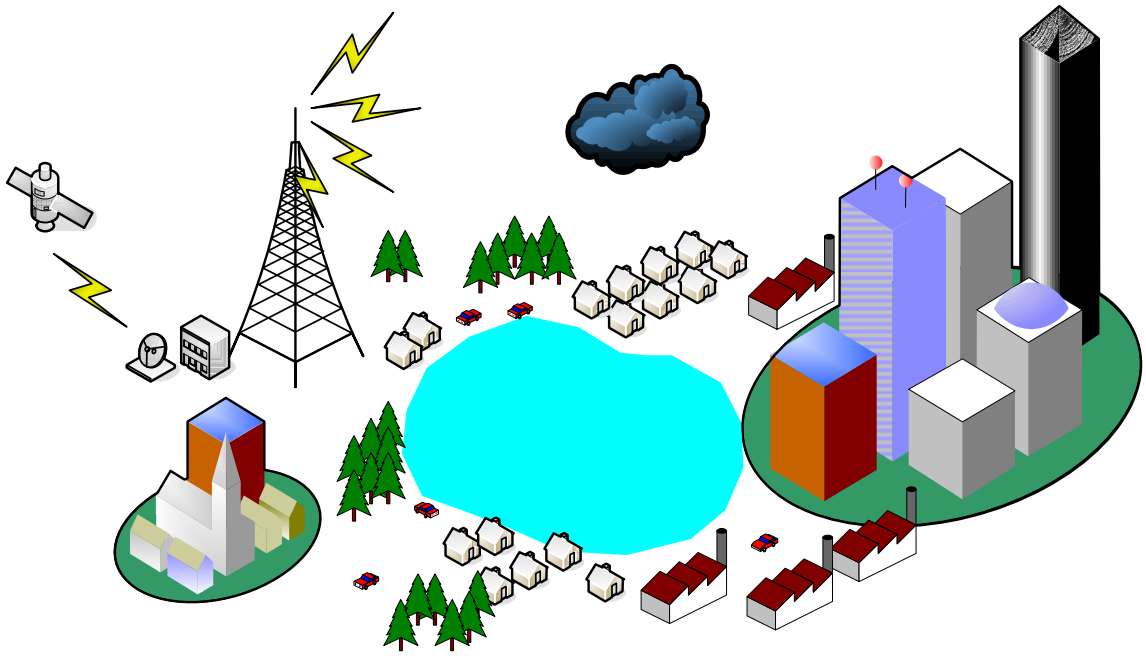

Figura 2.1: Ilustração representando um possível cenário de transmissão de dados em b*roadcast.*

Um sistema de comunicação digital, como o de TV digital, tem algumas funcionalidades que são consideradas básicas, cabendo ao desenvolvedor de cada sistema acrescentar outras etapas que venham a melhorar o desempenho, ou com finalidades que

fogem ao escopo deste trabalho. Um diagrama contendo os blocos básicos de um sistema desta natureza está representado na Figura 2.2.

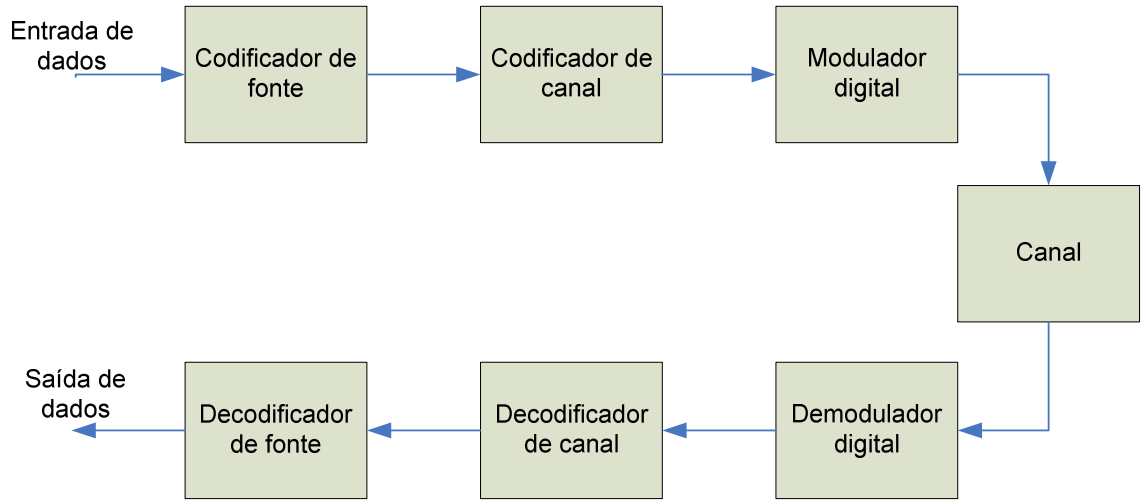

Figura 2.2: Diagrama de blocos representando os elementos básicos de um sistema de comunicação digital, ordenados de acordo com o fluxo dos dados (setas azuis).

A entrada de dados, conforme representada na Figura 2.2, é uma fonte de sinal digital ou analógico. No caso do sinal analógico, que pode ser proveniente de uma fonte de sinal de áudio ou vídeo, este deverá ser convertido para o formato digital, através de um processo de amostragem que permita que as informações contidas no sinal possam ser reproduzidas no receptor.

Os dados binários (digitais) são arranjados de forma a ocuparem a menor quantidade de bits possível para representar as informações provenientes da fonte de dados. Este arranjo é na maioria das vezes um algoritmo de compactação. Em sistemas de transmissão de televisão digital, o sinal de vídeo digital, ou analógico convertido para digital, é compactado, na maior parte dos casos, no padrão MPEG-2, e em sistemas mais modernos no padrão MPEG-4.

Uma vez arranjados os dados de maneira adequada, eles são passados por um bloco conhecido como codificador de canal (Figura 2.2) [1][2][3][4]. Este bloco é responsável pela inserção de dados de redundância que possibilitem, no receptor, recuperarem-se os dados enviados que tenham sido corrompidos pela ação degradante do canal. Este bloco pode conter vários outros sistemas de codificação de sinal, não necessariamente voltados à correção de erros, que terão o seu dual no receptor. Mais comumente utilizam-se dois sistemas voltados à correção de erro, o Reed-Solomon e a codificação convolucional. É sobre este último o foco do presente trabalho. Ainda no bloco de codificação de canal,

dentre inúmeras outras possíveis funções referentes a cada sistema em particular, é usual encontrarem-se mecanismos de byte e bit *interleaving*. A utilidade destes mecanismos e a forma com que se relacionam com o tema proposto para este trabalho serão tratadas mais adiante.

A seqüência binária da saída do codificador de canal é aplicada na entrada do "modulador digital" [2] [3], representado pelo bloco de mesmo nome na Figura 2.2. A função do modulador é mapear o sinal digital binário em uma forma de onda que possa ser propagada através do canal. No contexto do projeto de um sistema de comunicação destinado à transmissão de sinais de TV digital terrestre, o canal é o espaço entre as antenas transmissora e receptora – o ar – através do qual a onda eletromagnética se propaga transportando as informações. Em sistemas desta natureza, a onda eletromagnética, de característica senoidal, é um sinal de RF continuamente gerado em uma freqüência que não pode variar além de uma estreita banda passante, sinal que é denominado portadora. O mapeamento é feito alterando-se uma ou mais características da portadora, conforme a seqüência de dados digitais, vindos do codificador, a serem enviados pelo transmissor. A modulação que utilizamos neste trabalho é conhecida como *M*-QAM, onde M determina o número de possíveis diferentes formas de onda de energia finita que a portadora poderá assumir. Cada uma destas formas de onda é chamada de símbolo, sendo b o número de bits que um símbolo pode representar, determinando, portanto, o valor de M como sendo $M = 2^b$.

O decodificador, cuja implementação é tratada neste trabalho, baseou-se em um sistema de modulação que utiliza $M = 16$ (16-QAM) ou $M = 64$ (64-QAM), em que cada símbolo é constituído por 4 bits ou 6 da seqüência de dados a serem transmitidos, respectivamente. A sigla QAM significa *Quadrature Amplitude Modulation*, modulação em que cada símbolo é representado por um número complexo $s = I + jQ$, onde o módulo de s representa a amplitude do sinal de RF e o ângulo de s representa a sua fase. Na Figura 2.3 está representada graficamente a constelação (conjunto dos possíveis símbolos) da modulação 16-QAM, sobre o espaço formado por um eixo cartesiano representando a dimensão real (*I*) e outro representando a dimensão imaginária (*Q*). Os círculos vermelhos representam as possíveis posições dos símbolos. Sobre os símbolos encontram-se, em amarelo, os valores binários que eles representam e, em azul, os valores hexadecimais.

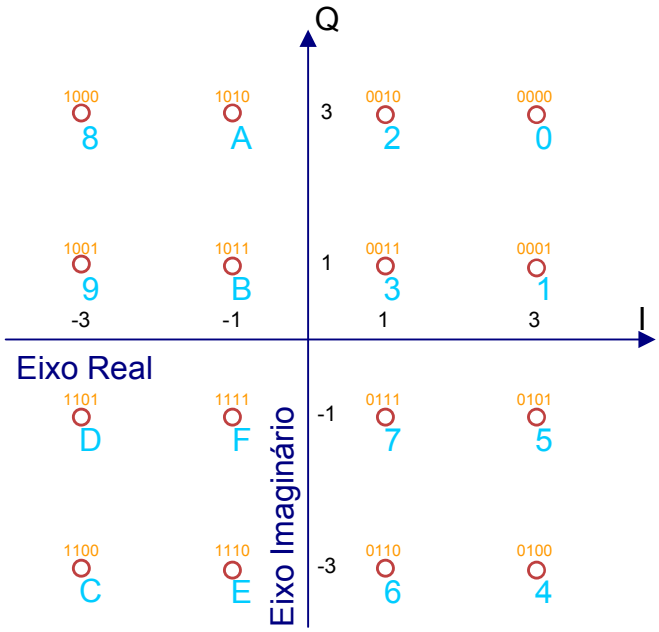

Figura 2.3: Representação gráfica da constelação resultante da modulação 16-QAM usada em sistemas de transmissão de TV digital, tais como DVB-T e ISDB-T.

O mapeamento do conjunto de bits que cada símbolo da constelação irá representar é de livre escolha do projetista do sistema de comunicação digital, bastando que este mapa seja conhecido no receptor, para que os dados possam ser recuperados à medida que cada símbolo é recebido. Porém, há formas de arranjar estes símbolos no mapa que tornam o sistema mais eficiente e menos sujeito a erros. Como é possível observar na Figura 2.3, o valor binário, que cada símbolo está representando, varia de apenas um bit entre os símbolos adjacentes entre si. A esta forma de distribuir os dados chama-se de Código Gray. Uma vez que a maioria dos erros, causados pelo ruído gaussiano imposto pelo canal, ocorrem em virtude do receptor entender um símbolo recebido como sendo um de seus adjacentes, a utilização do Código Gray permite que este erro resulte em apenas um bit de erro na seqüência de dados recebida.

Mesmo utilizando-se do código Gray, há mais de uma forma de mapeamento dos símbolos na constelação. Neste trabalho utilizamos o mapa representado na Figura 2.3 por permitir-nos dividir a constelação em quadrantes e sub-quadrantes. A forma que esta divisão é feita e a razão de sua escolha serão tratadas posteriormente neste trabalho.

Após a modulação, a portadora terá sua freqüência elevada até um valor que corresponda à freqüência reservada para a transmissão deste sinal específico de TV digital. O sinal de RF resultante será amplificado permitindo que ele seja difundido, através de

uma antena, pelo ar (o canal), cobrindo a área de alcance da transmissão. O canal [2] [3] é um dos blocos representados na Figura 2.2 e é o elo entre o transmissor e o receptor.

Em se tratando do meio atmosférico, estamos trabalhando com um dos meios de propagação mais hostis, no sentido de possíveis fontes de interferência, contínuas e impulsivas, inserção de ruídos aditivos, atenuação do sinal, pontos de reflexão e outros tipos de distorções.

O ruído aditivo gaussiano é uma das principais fontes de degradação do sinal. Este ruído geralmente é oriundo de duas principais fontes, sendo a principal a variação das características, imposta pela variação térmica, dos componentes eletrônicos e amplificadores da etapa de RF dos receptores. A segunda fonte mais comum de ruído aditivo gaussiano são radiações interferentes em freqüências que passem pelas etapas de filtragem do receptor de RF. Um modelo deste canal pode ser represento pelo diagrama da Figura 2.4, onde $s(t)$ é o sinal transmitido e $r(t)$ é este sinal depois de passado pelo canal e corrompido pela adição do ruído aleatório $n(t)$. Este ruído é uma das partes do chamado "Canal sem memória", devido a sua característica aleatória e independente de eventos antecedentes ou posteriores.

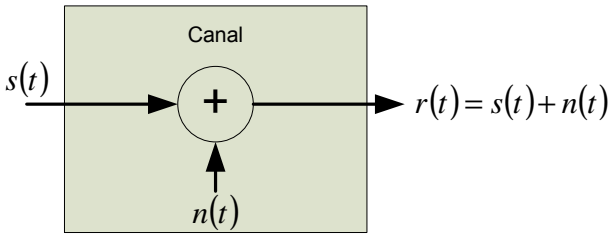

Figura 2.4: Representação do ruído aditivo aleatório do canal.

Também são parte do "Canal sem memória" os ruídos impulsivos, que são interferências causadas momentaneamente e sem qualquer relação causal com outros eventos interferentes do canal. As fontes mais comuns deste tipo de interferência são faíscas, como raios, ou as induzidas nas velas em motores a explosão interna e etc.

As interferências com memória, isto é, provenientes da convolução do sinal de RF ao longo do tempo com as características de distorção do canal, são eliminadas através de um dispositivo chamado de equalizador que se encontra no receptor. O equalizador, através de um modelo, estático, ou atualizado dinamicamente, utiliza-se da função de transferência inversa do canal de transmissão para desconvoluir o sinal. Uma vez que nem o equalizador, nem os efeitos do "Canal com memória", que ele anula, estão relacionados diretamente

com a etapa de codificação e decodificação convolucional, este tema não será tratado neste trabalho. Apenas é importante registrar que após a etapa de equalização o sinal resultante é, idealmente, um sinal com interferências "puramente" aditivas gaussianas e impulsivas.

Em algum lugar no raio de alcance do transmissor um suposto receptor através de uma antena imersa no ar (canal) recebe o sinal de RF. O sinal é tratado e trazido para a banda base (mesma freqüência e modulação que tinha na saída do modulador) na etapa de RF. Este sinal é entregue ao demodulador [1][2][3] onde será amostrado por um conversor AD. Os dados amostrados no AD são inseridos nos sistemas de sincronismo de portadora e sincronismo de símbolos [2], que permitirão distinguir o início e o fim dos símbolos no tempo. Além dos sincronismos há também o equalizador já citado anteriormente.

Os sinais digitais na saída do equalizador são o mais próximo possível dos sinais na saída do mapeamento de símbolos no transmissor, diferindo apenas pelo espalhamento em relação às posições de referência, conseqüência do ruído aditivo gaussiano. A etapa de "demapeamento" , neologismo que indica a atividade de procurar no mapa de símbolos a posição mais próxima da qual o símbolo recebido se encontra, é, via de regra, considerada parte integrante do demodulador. Neste trabalho consideraremos o "demapeamento" como parte do decodificador de canal, que, embora não seja usual, está de acordo com a interatividade necessária entre "demapeador" e o decodificador de Viterbi com *Soft-Decision*. Estes, acabam por dividir a tarefa de decidir a suposta posição original do símbolo que foi recebido e, portanto, os bits que o representam.

Após o bloco do demodulador, conforme o diagrama da Figura 2.2, segue o decodificador de canal, bloco responsável por desfazer toda a codificação feita no codificador de canal do transmissor. Fora as diversas codificações que possam ocorrer por características particulares de cada diferente sistema, no caso de sistemas de TV digital, é usual estarem presentes o decodificador convolucional, responsável pela correção de erros em bits e o decodificador *Reed-Solomon* que é responsável pela etapa de correção de erros em blocos (geralmente bytes). Este trabalho trata especificamente do decodificador convolucional de Viterbi usando o método *Soft-Decision* para a decodificação, antecedido de um bit *deinterleaver*.

Depois de todas as etapas da transmissão e da recepção digital de dados, finalmente os bits chegam ao decodificador de fonte, onde são rearranjados no seu formato original, quando da entrada no processo de transmissão digital, sendo possível recuperar o sinal digital ou analógico. Este sinal, no caso da TV digital, será então conduzido à entrada de

um aparelho de televisão onde as imagens e sons transmitidos pela emissora serão exibidos.

2.1 Codificador Convolucional

2.1.1 O Codificador

O codificador convolucional [1][2][3][4] é um processo destinado à correção de erros em bits da seqüência de dados transmitidos em um sistema de comunicação digital. Este processo destina-se a corrigir principalmente os erros ocasionados pelo ruído aditivo gaussiano, ou por algum ruído impulsivo no canal. Basicamente o codificador convolucional tem o objetivo de mudar artificialmente a característica de "falta de memória" do canal aditivo gaussiano, gerando uma dependência do conteúdo do dado enviado com o conteúdo dos seus dados sucessores e antecessores. O cumprimento desta tarefa é feito passando-se os dados a serem enviados por uma máquina de estados finitos e com um numero de possíveis transições entre estados também finitos. Ao se decodificar, no receptor, os dados enviados pelo transmissor, é possível detectar a ocorrência de transições não previstas no projeto do codificador, indicando assim a ocorrência de um ou mais erros na seqüência de dados recebidos. Como veremos mais adiante neste trabalho, o decodificador convolucional, através do "conhecimento" das transições entre estados que são permitidas no codificador, é capaz de corrigir eventuais dados corrompidos da seqüência recebida.

Um codificador convolucional é composto por uma memória tipo *shift register* de K estágios, cada estágio composto de k bits. Esta memória é parte integrante de uma máquina de estados finitos, que tem sua saída condicionada por operações algébricas aplicadas sobre o conteúdo desta memória. O número K de estágios da memória do codificador é chamado de *constraint length*. Os dados binários são aplicados, k bits por vez, na entrada do codificador e deslocados sobre as posições da memória, descartando-se os k bits mais antigos. Consideraremos que o número de bits que saem do codificador, para cada k bits que entram, seja representado por n. A taxa de codificação T_C é dada por $T_C = k/n$.

O conteúdo do *shift register* é, na maioria das vezes, iniciado com um valor conhecido. A partir daí, considerando as infinitas possíveis seqüências de dados a serem aplicadas na entrada do codificador, há infinitas possibilidades de seqüências de saída do mesmo.

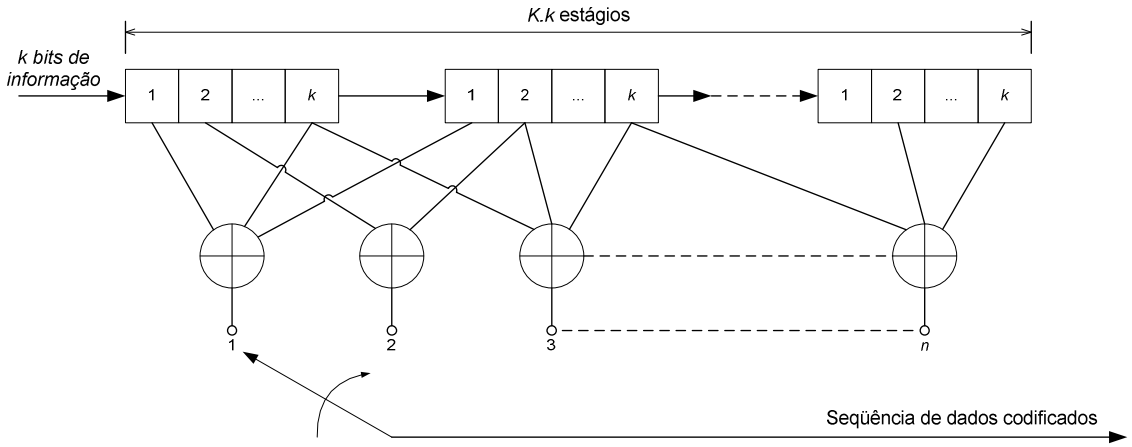

Figura 2.5: Diagrama de blocos de um codificador convolucional genérico.

A Figura 2.5, assim como o texto inicial deste capítulo, são baseados no livro de John G. Proakis [2]. A Figura 2.5 apresenta um modelo genérico de um codificador convolucional, através de diagrama de blocos de funcionalidade equivalente, ou também chamado de representação de conexões. Neste diagrama estão representados os K blocos de memória contendo k bits cada, interligados aos n operadores lógicos "ou exclusivo". Os dados da saída do codificador são organizados de forma serial através de uma chave rotatória que sequencialmente seleciona os n bits na saída dos n operadores lógicos, formando a seqüência serial codificada.

A partir deste ponto será utilizado um exemplo simples (retirado do livro de B. Sklar [3]) de um possível codificador convolucional, para que se torne mais simples a exposição dos conceitos envolvidos nesta codificação. O referido exemplo consiste de um codificador convolucional de $K = 3$ posições de memória contendo $k = 1$ bits cada uma. Serão usados $n = 2$ operadores lógicos do tipo "ou exclusivo", resultando numa taxa de codificação $T_C = k/n$ de ½. O diagrama de conexões que representa este codificador está ilustrado na Figura 2.6, sendo m a seqüência binária submetida ao codificador e U a saída do mesmo, composta por u_1 e u_2 ordenados serialmente.

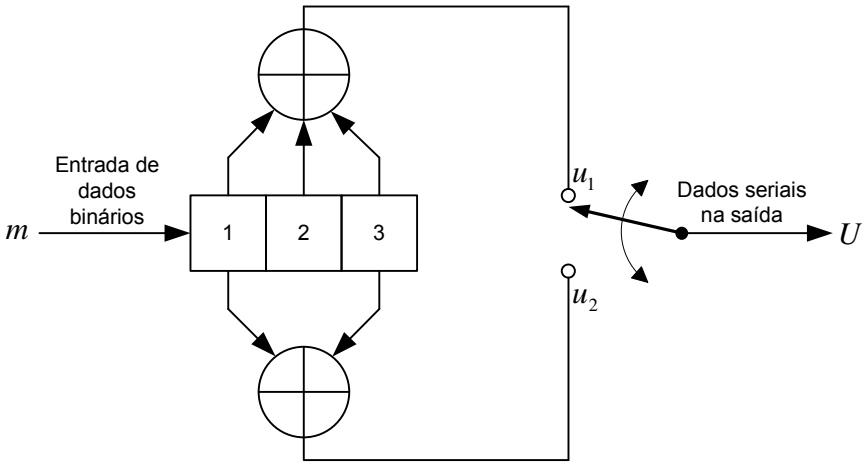

Figura 2.6: Codificador convolucional com taxa de codificação de ½ e K = 3.

As conexões dos operadores lógicos com a memória podem ser expressas utilizando a forma vetorial. Esta representação utiliza vetores binários para descrever as conexões com cada um dos *n* operadores lógicos do codificador. No exemplo da Figura 2.6 um vetor de conexões superior g_1 e um vetor de conexões inferior g_2 descrevem as conexões da memória com os dois operadores lógicos. A posição do elemento do vetor, correspondente a um dos K elementos da memória, ocupada pelo valor zero, denota que nesta posição não há conexão entre a memória e o operador lógico. A presença do valor "um" indica a existência de ligação. Portanto os vetores correspondentes às ligações deste codificador são:

$$g_1 = 111$$

$$g_2 = 101$$

É bastante usual as seqüências de bits que formam os vetores de ligações do codificador serem escritas em formato octal. Neste caso, se diria que o codificador apresentado na Figura 2.6 é gerado por:

$$G_1 = 7 \text{ Octal}$$

$$G_2 = 5 \text{ Octal}$$

Em muitas situações o codificador convolucional é representado por n polinômios geradores, sendo cada polinômio a representação das conexões de um dos n operadores lógicos da máquina de estados com as posições do *shift register*. Cada polinômio será de um grau $K-1$ ou menor. Os coeficientes de cada um dos termos do polinômio de ordem $K-1$ serão 1 quando o termo corresponder à uma posição em que haja conexão entre a

memória e o operador lógico, ou 0 quando não houver conexão. De forma muito similar à representação por vetores de conexões, $g_1(X)$ é o polinômio que representa as conexões superiores e $g_2(X)$ representa as conexões inferiores. Assim teremos para o exemplo do codificador da Figura 2.6:

$$g_1(X) = 1 + X + X^2$$

$$g_2(X) = 1 + X^2$$

onde o termo de menor ordem representa o estágio de entrada de dados do *shift register*.

Uma outra forma de representação do codificador convolucional bastante comum, porém viável apenas para codificadores de pequena complexidade, é o diagrama de estados. Observando-se o codificador da Figura 2.6, é possível perceber que quando um novo bit entra na primeira posição do *shift register* em um instante de tempo t e o bit da última posição é descartado, apenas o bit que entrou e os dois bits das posições mais a esquerda da memória, no instante $t-1$, é que influem no valor assumido na saída, assim como no novo conteúdo da memória. Portanto pode-se dizer que a saída do codificador é decorrente do bit que entra mais os quatro possíveis estados definidos pelos dois primeiros bits do *shift register*.

O diagrama de estados do codificador apresentado na Figura 2.6 é ilustrado na Figura 2.7, onde o conteúdo das caixas é o estado atual da máquina de estados. Neste diagrama há setas representando as possíveis transições entre os estados, junto a elas há um texto no formato $B/u_1\,u_2$, onde B é o bit que entrou e u_1 e u_2 os bits resultantes na saída do codificador.

A representação do codificador convolucional por diagrama de estados torna mais claro o seu funcionamento, porém não é possível mapear o histórico das transições de estado ao longo do tempo. Este mapeamento pode ser feito de forma bastante simples utilizando-se da representação das mudanças de estados do codificador em um diagrama de árvore de estados, ilustrado na Figura 2.8. Neste tipo de diagrama, cada forquilha representa um estado do codificador e cada ramo uma possibilidade da entrada de um bit de valor 1 ou 0. Sobre os ramos das forquilhas estão representados o bit B que entrou e os dados u_1 e u_2 na saída do codificador, no formato $B/u_1\,u_2$ (em preto). Na base de cada forquilha estão representados, por uma letra em azul, cada um dos quatro possíveis estados, onde $a = 00$, $b = 10$, $c = 01$ e $d = 11$. A vantagem da adição da dimensão do tempo é a possibilidade de observar e mapear a codificação do sinal a partir de cada diferente dado

que entra. A desvantagem é que em poucos números de iterações já não será possível representar o processo em uma folha de papel.

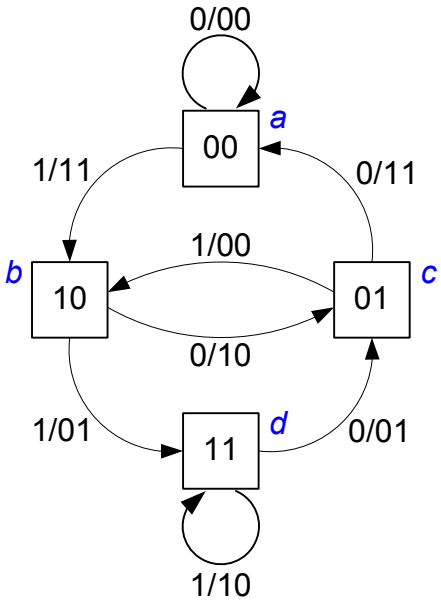

Figura 2.7: Representação do diagrama de estados do codificador convolucional da Figura 2.6.

No exemplo de representação de árvore de estados da Figura 2.8, há uma marcação em vermelho sobre alguns ramos da árvore. Esta marcação representa o caminho percorrido na árvore para o caso de ser apresentada uma seqüência de dados m={1 0 0 1 1} na entrada do codificador. Como é possível observar na figura, esta seqüência gera uma saída com valores U = {11 10 11 11 01}.

Uma forma alternativa para a representação de um codificador convolucional, acrescentando a dimensão do tempo, porém sem o inconveniente da área necessária para desenhar o diagrama de árvore de estados, é através do diagrama de treliça. Este tipo de diagrama é largamente utilizado por, além das razões já citadas, ser altamente compatível com a maneira que o decodificador de Viterbi trata a decodificação convolucional, conforme veremos mais adiante neste trabalho. Assim como o diagrama de estados, o diagrama de treliça também se torna mais difícil de ser representado quanto maior o número de possíveis estados do codificador.

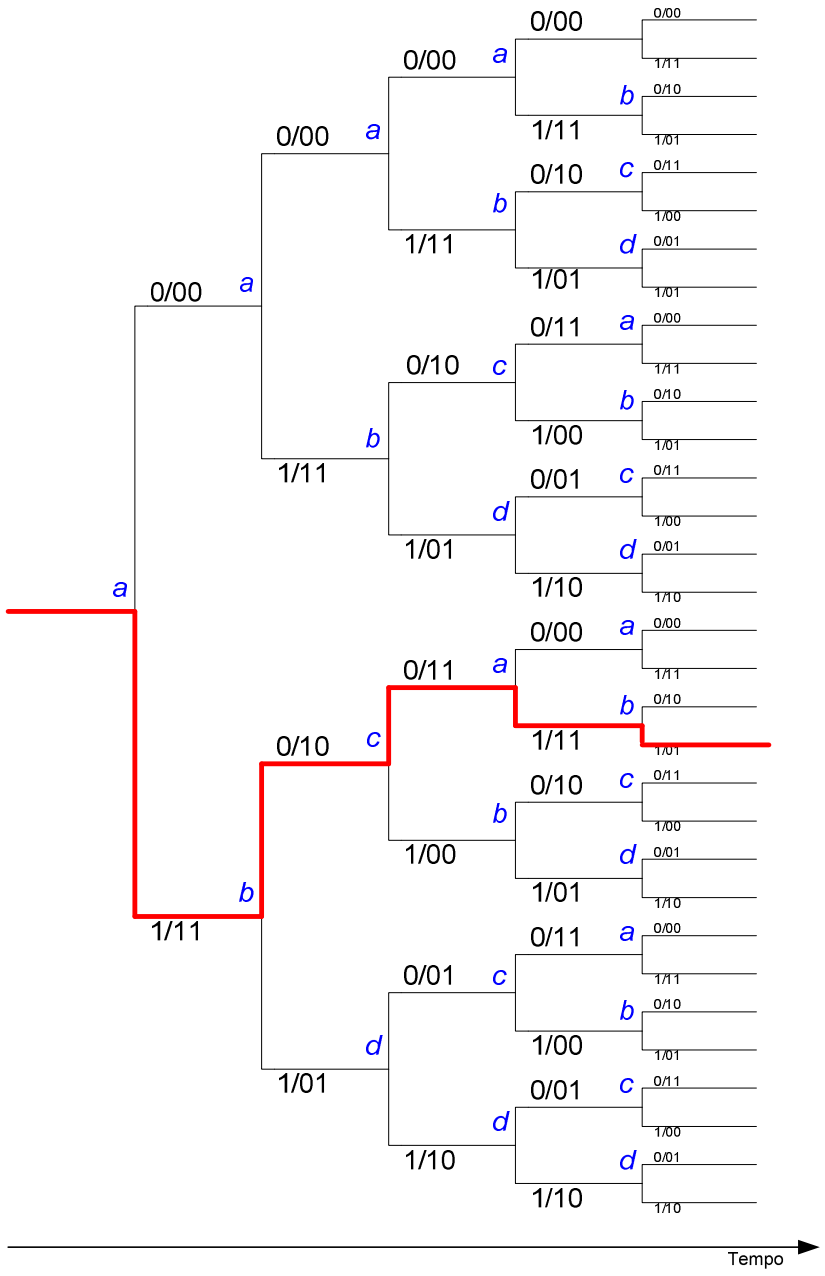

Figura 2.8: Exemplo de representação em árvore do codificador apresentado na Figura 2.6.

Pode-se dizer que um diagrama de treliça é uma maneira otimizada de representar o diagrama de árvore de estados. Observando-se a Figura 2.8 pode-se perceber que a estrutura da árvore repete-se a cada três estágios. Isto é caracterizado pelo fato de o *constraint length* ser $K = 3$. O diagrama de treliça focaliza apenas na representação das possíveis transições entre os possíveis estados do codificador convolucional. Na Figura 2.9 há o desenho de um diagrama de treliça correspondente ao codificador convolucional da Figura 2.6, em que cada linha de pontos representa um dos possíveis estados. Cada coluna

de pontos é o conjunto de todos os possíveis estados em diferentes instantes de tempo, com as setas representando as possíveis transições entre os estados. Neste gráfico há uma coluna de valores em verde à esquerda que representam os estados que correspondem a cada linha de pontos. Junto a cada uma das setas estão escritos valores em azul representando o bit que entrou no codificador B e os dados de saída do codificador (u_1 e u_2) no formato $B/u_1\ u_2$.

Na treliça do exemplo representado na Figura 2.9, há uma marcação em vermelho em algumas transições. Esta marcação indica o caminho percorrido no caso de ser apresentada uma seqüência de dados m={1 0 0 1 1}na entrada do codificador. Como é possível observar na figura, esta seqüência gera uma saída com valores U = {11 10 11 11 01}.

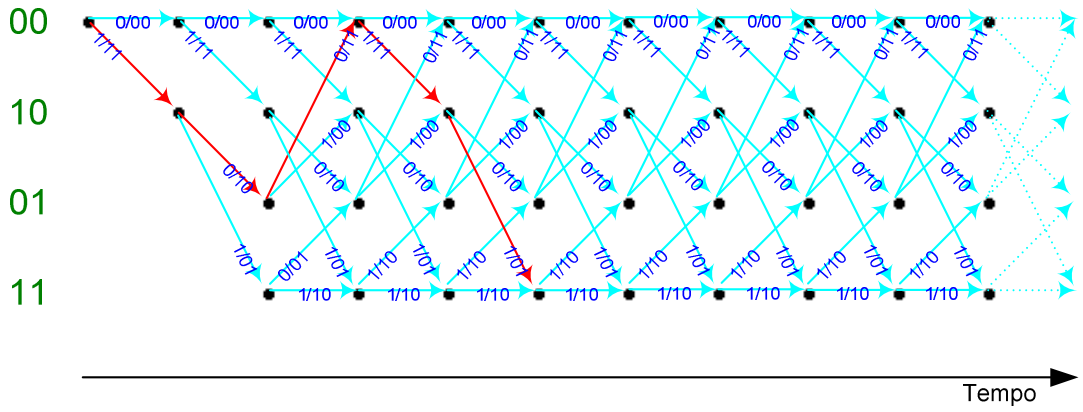

Figura 2.9: Representação por diagrama de treliça para o codificador apresentado na Figura 2.6.

2.1.2 *Puncturing*

Em algumas aplicações faz-se necessário que os codificadores convolucionais tenham altas taxas de codificação, para permitir o transporte de uma maior quantidade de dados úteis durante a etapa de transmissão. Uma vez que, em um codificador com uma taxa de codificação $T_C = 1/2$, tenham-se dois bits saindo para cada bit que entra, temos na saída um código em que apenas 50% do volume de dados são informações referentes ao conteúdo que se deseja transmitir, os 50% restantes são informações redundantes que são

enviadas junto, exclusivamente para auxiliar na correção de eventuais erros nos dados recebidos no receptor.

É possível aumentar a taxa de codificação de um codificador convolucional sem que seja necessário alterar a estrutura e o funcionamento do mesmo. Para tanto, após a etapa de codificação utiliza-se de um mecanismo que faz o processo de *puncturing* (puncionar) nos dados da saída do codificador. Este processo consiste em eliminar, de forma periódica e criteriosa, bits da seqüência de dados da saída do codificador convolucional. Assim se tivermos um codificador com taxa de codificação $T_C = 1/2$, para cada três bits aplicados na entrada, teremos seis bits na saída, eliminando-se dois destes bits da saída então teremos uma taxa $T_C = 3/4$. Isto significa que 75% do volume total de dados transporta informação útil e apenas 25% transporta informação redundante necessária ao processo de identificação e correção de erros. Esta técnica utiliza-se da capacidade de correção de erros do decodificador para garantir que os dados extraídos da seqüência binária serão recuperados no receptor. As vantagens do *puncturing* se contrapõem ao fato de a capacidade de correção do decodificador diminuir quanto maior a taxa de codificação obtida por este processo.

A descrição do processo de *puncturing,* que deve ser aplicado no conjunto de dados saídos do codificador convolucional, é feita em forma de uma matriz. Esta matriz terá tantas linhas quanto o número de posições que a chave rotatória da saída da máquina de estados tiver. O número de colunas desta matriz é o número de ciclos que a chave rotatória da saída do codificador terá de executar para que a seqüência de *puncturing* se repita. A matriz de *puncturing* ideal é aquela que gera a taxa de codificação desejada para o processo de codificação convolucional, comprometendo ao mínimo a capacidade de correção de erros do decodificador. Estas matrizes "ideais", assim como as estruturas dos próprios codificadores convolucionais, na maioria das vezes são obtidas através de extensas buscas computacionais.

O funcionamento do *puncturing* pode ser descrido como sendo diretamente ligado ao funcionamento da chave de saída da máquina de estados do codificador convolucional. Se o valor contido na matriz, na linha correspondendo à saída e na coluna correspondente ao estágio do ciclo de chaveamento em que a chave se encontra for zero, então a chave deve descartar o bit da saída e comutar para a posição seguinte. Para ilustrar, utilizando-se

do exemplo de codificador da Figura 2.6, consideraremos uma seqüência de saída hipotética em u_1 e u_2:

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Estes valores de u_1 e u_2 geram uma saída do codificador U={1 1 1 0 1 1 1 1 0 1 1 0}.

Utilizaremos a matriz de *puncturing* [1][2] para que o codificador do exemplo da Figura 2.6 funcione com uma taxa de codificação $T_C = 3/4$ em lugar da original $T_C = 1/2$, apresentada a seguir:

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

Após o *puncturing* a saída do codificador será U={1 1 0 1 1 1 1 1}. Para que estes dados possam posteriormente serem decodificados pelo decodificador de Viterbi, no receptor digital, será necessário passá-los pelo processo de *depuncturing*. O processo de *depuncturing* será tratado no capítulo sobre o decodificador convolucional de Viterbi.

2.1.3 Mínima distância livre de um código convolucional

Quando falamos em codificação e decodificação convolucional, fala-se sempre em distâncias. Há dois tipos distintos de distâncias as quais podemos nos referir, a distância de Hamming e a distância Euclidiana. Por hora trataremos da distância de Hamming, que é basicamente o número de elementos que duas palavras binárias (código) diferem entre si, ou o número de posições em que elas diferem. Conforme veremos mais adiante, as distâncias são a base de todo o funcionamento do mecanismo de correção de erros feita utilizando-se o código convolucional.

A mínima distância livre de um código convolucional é um indicativo da "qualidade" de um código, ou de sua eficiência para correção de erros. Quanto maior a mínima distância livre maior a eficiência. Quanto menor a mínima distância livre, mais fácil será um código corrompido ser aceito como um código válido, pois poucos bits invertidos por ação de ruído e/ou interferência, seguidos um logo após o outro, podem formar uma seqüência que também seria oriunda de uma transição válida no codificador. A utilização da técnica de *puncturing* diminui a mínima distância livre do código, portanto há

um compromisso a ser obedecido ao se projetar um sistema de comunicação digital, levando-se em consideração a taxa de transmissão de dados úteis desejada e a capacidade de correção de erros necessária.

A obtenção da medida da mínima distância livre de um código pode ser feita traçando-se dois caminhos distintos sobre a treliça a partir de duas transições que partam de um mesmo estado. Estes caminhos devem ser traçados de forma a se encontrarem novamente em um mesmo estado após o menor número possível de transições. O número de bits que diferirem entre as seqüências dos dados de saída U representadas por cada um dos caminhos será a mínima distância livre. Na Figura 2.10 há o exemplo de vários pares de caminhos (cada par é representado por setas de uma mesma cor) que podem ser usados para determinar a mínima distância livre do código do codificador representado na Figura 2.6. A mínima distância livre, deste exemplo, é cinco, e será encontrada a mesma distância livre para quaisquer caminhos que se possa escolher sobre esta treliça, devido à propriedade linear dos códigos convolucionais (caso mantidos os critérios citados anteriormente para traçar tais caminhos).

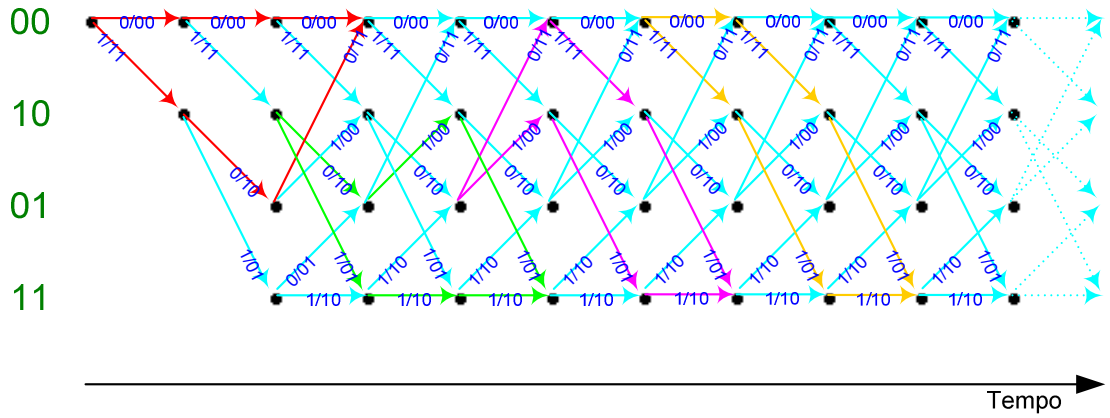

Figura 2.10: Exemplos de pares de caminhos traçados em diferentes cores para obtenção da característica de mínima distância livre do codificador apresentado na Figura 2.6.

2.2 Bit *interleaving* e bit *deinterleaving*

Conforme citado anteriormente, uma das características do canal que geram problemas de inserção de erros na seqüência binária enviada do transmissor ao receptor de um sistema de telecomunicações digitais, são os *bursts* de ruído impulsivo. Muitas vezes

este tipo de ruído causa um número tão grande de erros subseqüentes na seqüência de bits recebidos pelo receptor, que esta quantidade excede a capacidade de correção de erros do decodificador convolucional. Não raramente quando o decodificador convolucional não consegue corrigir uma seqüência de bits corrompidos, a decodificação errônea propaga-se ao longo da treliça do decodificador, causando mais erros do que os originalmente introduzidos pelo canal de características aditivas gaussianas e impulsivas.

O decodificador convolucional tem a sua capacidade de correção de erros significativamente aumentada quando os erros ocorrem espaçados uns dos outros em comparação com seqüências em que venham grupos de bits corrompidos. Para garantir o funcionamento do decodificador convolucional em uma situação o mais próxima possível da ideal, após a etapa de codificação convolucional há uma etapa de bit *interleaving*. Nesta etapa os bits são ordenadamente redistribuídos no tempo, de forma que bits originalmente próximos passem a ficar o mais distante possível entre si.

No receptor, antes dos dados serem decodificados pelo decodificador convolucional, há o processo de *deinterleaving*. Neste processo é feita a reorganização dos bits para que eles voltem a ficar em suas posições originais (anteriores ao *interleaving* feito na etapa de transmissão). Desta forma os bits que estavam próximos durante a passagem dos dados pelo canal, serão distanciados uns dos outros. Este afastamento dos bits que foram transmitidos próximos uns dos outros, permite que esta seqüência binária, no caso de vários bits subseqüentes terem sido corrompidos por efeito de um *burst* de ruído impulsivo, seja decodificada com menores chances de exceder o limite da capacidade de correção de erros consecutivos do decodificador convolucional.

Uma representação gráfica, ilustrando o funcionamento do bit *interleaver,* é apresentada nas Figuras 2.11, 2.12, 2.13 e 2.14. Nestas figuras é apresentado um exemplo hipotético de um sistema de comunicação digital que tem os bits de dados (cubos numerados) a serem transmitidos organizados em blocos de 20 bits (os números dentro dos cubos representam a posição original que eles ocupavam no bloco de dados de sua origem, por exemplo, o cubo de numero 1 de cor verde é o primeiro bit do bloco verde, assim como o cubo azul de número 1 é o primeiro bit de um outro bloco, o de cor azul). Os dados são apresentados em sua forma original, logo após a codificação convolucional no transmissor digital, na Figura 2.11. Na Figura 2.12 está representado o mesmo bloco de bits após o bit *interleaving* feito por um suposto bit *interleaver* ainda no transmissor. Na Figura 2.13 aparece o bloco da Figura 2.12 com bits coloridos em vermelho representando bits

corrompidos por um *burst* de ruído impulsivo ocorrido durante o suposto transporte destes dados através do canal de transmissão (sendo o cubo vermelho de número 6 o 6º bit do bloco azul e os cubos vermelhos de números 5, 9 e 13, os bits 5º, 9º e 13º, respectivamente, do bloco de dados verde – todos em vermelho representando bits corrompidos por um *burst* de ruído impulsivo do canal). Finalmente na Figura 2.14 aparecem os bits rearranjados em sua forma original, pelo bit *deinterleaver* do receptor digital. Na última figura é possível observar que os bits corrompidos, que foram enviados através do canal em posições subseqüentes, estão distanciados entre si antes de serem remetidos à etapa de decodificação convolucional.

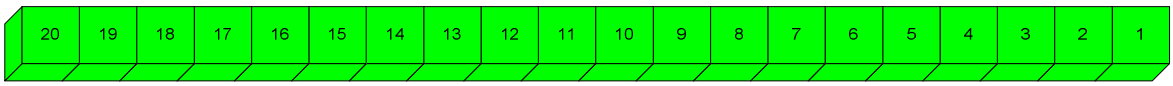

Figura 2.11: Seqüência de vinte bits saídos do codificador convolucional do transmissor digital e que será submetida ao bit *interleaver*.

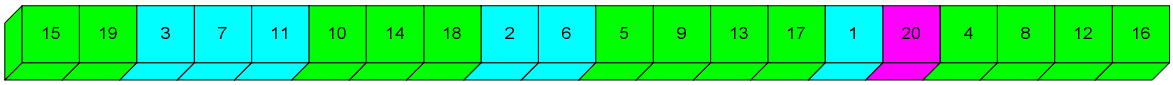

Figura 2.12: A seqüência de vinte bits apresentada na Figura 2.11, porém já passada pelo bit *interleaver*. Os cubos de cores diferentes representam bits dos blocos passados pelo *interleaver* antes (magenta) e depois (azul) do bloco deste exemplo.

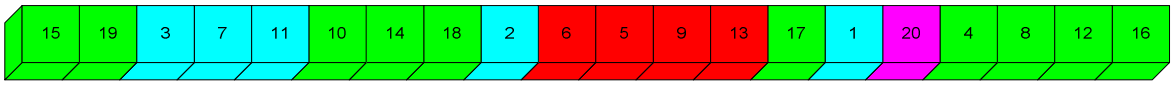

Figura 2.13: Esta figura representa os bits, corrompidos (em vermelho) pela ocasião da adição de ruído impulsivo, que chegam ao receptor após o envio do bloco de dados da Figura 2.12, através canal de transmissão.

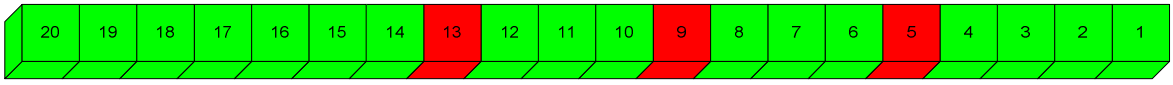

Figura 2.14: Representação da seqüência de dados da Figura 2.11 depois de serem reorganizados na sua forma originalmente saída do codificador convolucional. Em vermelho estão representados os bits que se encontravam juntos no momento em que os dados foram corrompidos durante a sua passagem pelo canal.

Há dois tipos principais de *interleavers*: O *interleaver* de bloco e o *interleaver* convolucional. Como os nomes sugerem o *interleaver* de bloco e o *interleaver* convolucional são mais comumente usados para condicionar os dados de codificadores de blocos e codificadores convolucionais, respectivamente. A codificação de blocos, assim

como a convolucional, é largamente utilizada em sistemas de telecomunicações digitais, mas não será tratada neste trabalho por estar fora do escopo do mesmo, da mesma forma que não serão tratados assuntos referentes ao *interleaving* de blocos.

O *interleaving* convolucional é considerado o tipo de intercalação ideal [2] para funcionar em associação com a codificação e a decodificação convolucionais. O *interleaving* convolucional é feito atrasando-se bits da seqüência de dados a serem transmitidos em relação aos demais bits, afastando-os dos seus bits adjacentes durante a etapa de transmissão através do canal. O tipo de interleaver convolucional de bits mais comumente utilizado é composto por um número M de memórias tipo *shift register* utilizadas como FIFOs (*First In First Out*) e dispostas em paralelo, conforme a ilustração da Figura 2.15. Cada uma destas FIFOs tem uma quantidade T de posições para armazenamento de bits a mais que a FIFO anterior. É usual adotar $T = 1$ como também adotar que a primeira "memória" não tenha nenhuma posição de armazenamento de bits (um condutor simples, portanto). As Entradas das FIFOs são "alimentadas" com bits, da seqüência de dados a ser transmitida, através de uma chave rotatória cíclica, que funciona em sincronismo com uma segunda chave, de mesmas características, que "recolhe" os bits na saída de cada uma das filas.

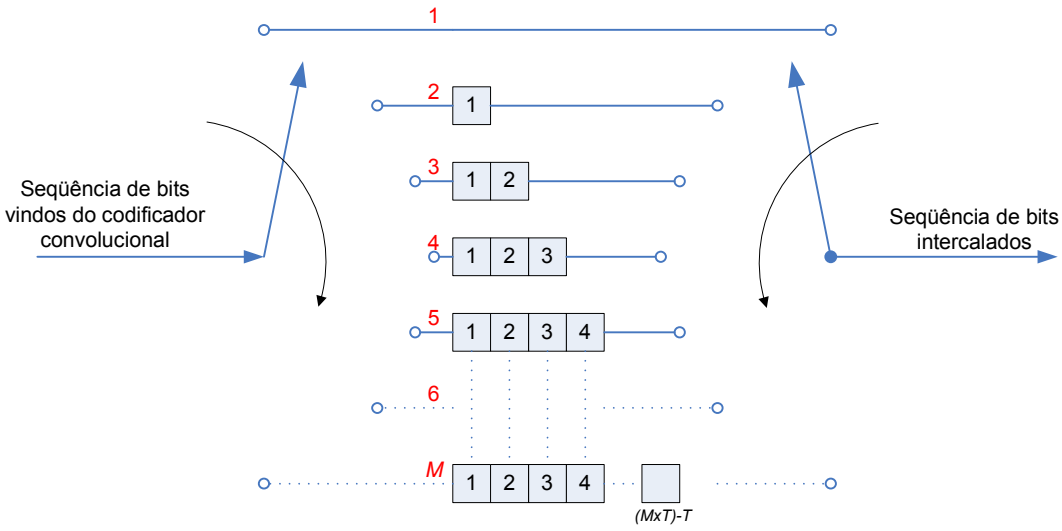

Figura 2.15: *Interleaver* convolucional de bits genérico com M FIFOs e $T = 1$ bits de diferença na quantidade de posições de memória, entre as FIFOs subseqüentes.

Como exemplo, consideraremos uma seqüência de dados organizados em blocos de 20 bits. Estes blocos serão passados por um bit *interleaver* semelhante ao apresentado na

Figura 2.15, com $M = 5$ e $T = 1$. Este exemplo condiz com o sistema que gerou o exemplo das Figuras 2.11, 2.12, 2.13 e 2.14, portanto podemos relacionar as informações de forma a considerar que o bloco de "dados verdes", apresentado na Figura 2.11 foi passado pelo bit *interleaver* deste exemplo, precedido por um bloco de "dados magenta" e sucedido por um bloco de "dados azuis". Na Figura 2.16, está representado o "escorregamento" causado pelo atraso imposto pela passagem dos dados pelas diferentes FIFOs, de diferentes tamanhos. Na Figura 2.12, está a imagem que ilustra como ficaria a seqüência de bits após serializados pela chave rotatória cíclica da saída do bit *interleaver*, considerando-se que "a imagem foi feita" no momento em que a chave de saída estava "recolhendo" o bit de número 16 do bloco de "dados verdes".

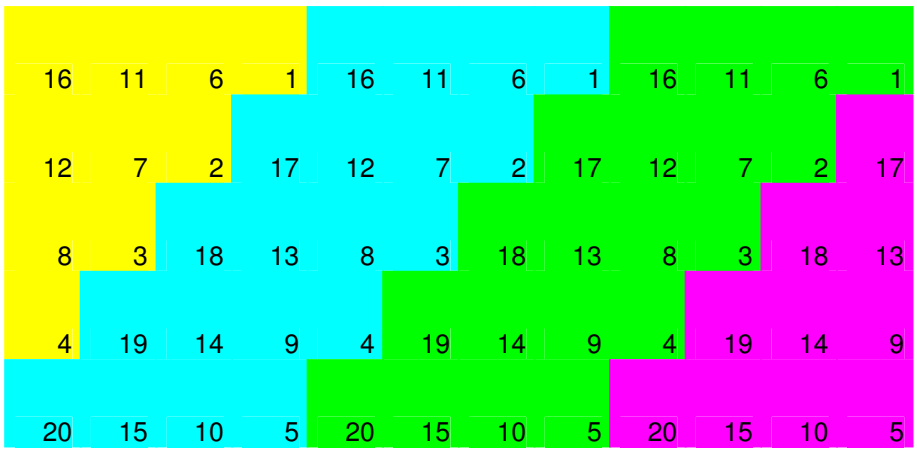

Figura 2.16: Representação do efeito de "escorregamento" causado nos conjuntos de dados passados pelo *interleaver* convolucional de bits. Cada uma das diferentes cores representa o grupo de bits pertencentes a cada um dos blocos de dados que entram seqüencialmente no bit *interleaver*. Os números nas diferentes posições da tabela representam a posição que cada um dos bits ocupava nos blocos de dados originalmente saídos do codificador convolucional.

Deste exemplo, vale a pena ressaltar o caráter meramente ilustrativo do mesmo. Em um sistema de comunicação digital com aplicação prática, muito provavelmente irá se desejar que os bits adjacentes da seqüência de dados a serem transmitidos que saem do codificador convolucional no transmissor, fiquem mais afastados entre si durante a etapa de transmissão sobre o canal. Há duas formas de se obter este maior afastamento e elas inclusive podem ser combinadas uma com a outra. A primeira maneira de aumentar-se o afastamento dos bits adjacentes após terem passado pelo processo de *interleaving*, é aumentando-se T, que geraria o resultado apresentado na Figura 2.17, no caso de $M = 5$ e $T = 2$, a ser comparado com o resultado mostrado na Figura 2.12, em que o codificador, também com $M = 5$, tem $T = 1$.

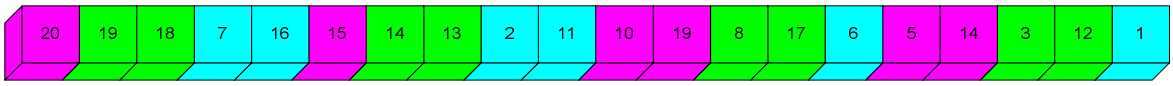

Figura 2.17: Seqüência de bits resultante da passagem, do bloco de bits representado na Figura 2.11, por um bit *interleaver* com $M = 5$ e $T = 2$.

A outra maneira de afastar mais os bits de uma seqüência de dados, a ser transmitida em um sistema de comunicação digital, através do uso de um bit *interleaver* convolucional, é aumentando-se o número de FIFOs M em paralelo. Da mesma maneira que foi feito com o exemplo da Figura 2.17, com base no exemplo da Figura 2.12, será mantido o valor de $T = 1$ e se alterará apenas o valor de M, que passará a ser $M = 10$, possibilitando uma comparação mais clara dos resultados referentes à alteração de apenas uma das características do *interleaver* de cada vez. O resultado do aumento de M pode ser observado na Figura 2.18, onde foi necessária a adição de duas novas cores em relação aos exemplos anteriores. A necessidade da adição de novas cores deve-se ao elevado grau de intercalação dos dados, que fez com que os bits se misturassem com bits de outros blocos ainda mais distantes no tempo, do que aqueles que tiveram partes representadas nos exemplos das Figuras 2.12 e 2.17.

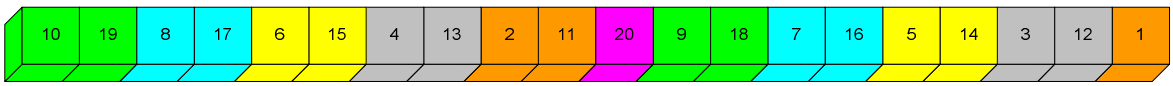

Figura 2.18: Seqüência de bits resultante da passagem, do bloco de bits representado na Figura 2.10, por um bit *interleaver* com $M = 10$ e $T = 1$.

O bit *deinterleaver* fica no receptor digital e é o mecanismo dual ao bit *interleaver* que fica no transmissor do sistema. A função do bit *deinterleaver* é reorganizar os dados recebidos, para que estes sejam entregues ao decodificador convolucional exatamente na mesma ordem que quando da sua saída do codificador convolucional. Para executar a tarefa de *deinterleaving*, basta ter no receptor um circuito ou um *software* que tenha as mesmas características e funcionamento que o sistema de *interleving*, porém as chaves rotatórias cíclicas devem funcionar em sentido de rotação oposto ao que faziam na distribuição e recolhimento dos bits nas FIFOs do bit *interleaver*. Desta forma os bits que sofreram maiores atrasos no *interleaver*, sofrerão menores atrasos no *deinterleaver* e vice versa, causando assim a neutralização do efeito causado pelo *interleaver*, e reagrupando os bits em seus blocos de origem, também recuperando a mesma ordem em que os dados saíram do codificador.

A principal característica da intercalação dos bits e da sua reorganização, após a passagem pelo canal, para o objetivo maior deste trabalho, que é a implementação de um decodificador convolucional de Viterbi com *Soft-Decision*, é o fato de bits subseqüentes serem transmitidos afastados uns dos outros através do canal. Como maior conseqüência desta característica está o fato de, por exemplo, em um sistema que utiliza modulação 16 QAM em que cada símbolo é formado por 4 bits da seqüência de dados digitais transmitidos através do ar, os bits que formariam um símbolo, na saída do codificador convolucional, não fazerem mais parte do mesmo símbolo a partir da etapa de *interleaving* e durante a etapa de propagação do sinal de RF no canal. Estes mesmos bits serão reunidos novamente após a etapa de *deinterleaving* que antecede o decodificador convolucional de Viterbi no receptor digital. Os problemas causados por esta característica serão discutidos mais além neste trabalho.

2.3 Decodificador Convolucional de Viterbi

O decodificador de Viterbi [1-19] é a forma mais utilizada na decodificação convolucional aplicada a sistemas de telecomunicações digitais, e muito provavelmente em todas as formas de "troca" de informações digitais binárias sujeitas à degradação imposta por qualquer tipo de canal aditivo gaussiano e/ou impulsivo, que venham a necessitar de um mecanismo de correção de erros que utilize codificação convolucional. Os maiores atrativos na escolha do decodificador convolucional de Viterbi como ferramenta de decodificação convolucional, voltada à correção de erro, são o seu ótimo desempenho, a leveza computacional e a baixa complexidade de *hardware* necessário para executar as rotinas de decodificação. A decodificação convolucional de Viterbi baseia-se na técnica de Decodificação por Máxima Verossimilhança (*Maximum Likelihood Decoding*), também usada para decodificação de dados codificados em blocos.

O funcionamento da Decodificação por Máxima Verossimilhança baseia-se em comparar a soma das distâncias de Hamming entre todos os dados da seqüência de dados recebidos no receptor digital e todos os dados de cada uma de todas as possíveis seqüências que podem ser geradas no codificador convolucional do transmissor digital. A menor soma destas distâncias indica a possível seqüência de dados que mais provavelmente teria sido gerada pelo codificador convolucional do transmissor. Graças à técnica de decodificação convolucional de Viterbi, é possível descartar a maior parte das

seqüências possivelmente geradas no codificador convolucional do transmissor e concentrar a pesquisa (comparação) apenas nas seqüências de dados com maior probabilidade de serem as que resultariam na seqüência de dados efetivamente recebidos no receptor digital. Devido a esta característica, o decodificador convolucional de Viterbi torna a decodificação convolucional por Máxima Verossimilhança viável de ser executada computacionalmente em sistemas de comunicação digital de alto desempenho.

O decodificador convolucional de Viterbi tem seu funcionamento baseado no conceito do diagrama de treliça, onde todas as possíveis transições entre todos os possíveis estados do codificador convolucional, localizado no transmissor digital, podem ser representados ao longo do tempo. O funcionamento de um decodificador convolucional de Viterbi será abordado a partir de um exemplo, que será apresentado a seguir.

Conforme citado anteriormente, um codificador convolucional é, na maioria das vezes, iniciado partindo-se de um mesmo estado pré-determinado a partir do qual o codificador funciona por um número fixo de bits a serem transmitidos antes que o mesmo seja reiniciado. Tal processo de inicialização-reinicialização sucessivas tem dois objetivos: (1) Permitir que o codificador (no transmissor digital) e o decodificador (no receptor digital) sejam sincronizados, tendo suas operações sempre iniciadas a partir de um mesmo estado comum entre ambos. (2) Evitar que erros em um grande número de bits "próximos" no tempo devido a um *burst* de ruído e/ou intereferência no canal de transmissão, acabem por causar uma decodificação errônea de seqüências de dados posteriores à ocorrência do *burst* (mesmo que estas estejam corretas) devido ao "estouro" da capacidade de correção do decodificador.

Como exemplo será considerado um sistema de comunicação digital *wireless*, tal qual em sistemas de TV digital terrestre, e que tenha em seu transmissor um codificador convolucional de mesmas características ao apresentado na Figura 2.6.

No desenvolvimento a seguir assume-se que:

- Os dados a serem transmitidos na entrada do codificador são representados pela seqüência de bits V .
- Os dados resultantes na saída da etapa de codificação convolucional no transmissor digital são representados pela seqüência de bits X .
- O estado inicial do codificador é o estado "00", também identificado como estado "a" na Figura 2.7.

- Os dados recebidos no receptor digital, após passagem pelo canal, são representados pela seqüência de bits Y.
- Os dados a serem obtidos na saída do decodificador convolucional de Viterbi serão representados pela seqüência de bits Z.

Consideraremos que:

V: 1 1 0 1 1 0 0 1 1 1 0

de onde se obtém, após a codificação pelo codificador da Figura 2.6:

X: 11 01 01 00 01 01 11 11 01 10 01

e para efeito de exemplo consideraremos a seqüência recebida no receptor como sendo a seqüência X com ocorrência de erro no décimo primeiro bit na order de ocorrência temporal (décimo primeiro bit da esquerda p/ direita):

Y: 11 01 01 00 01 **1**1 11 11 01 10 01
↑
erro

Neste exemplo será seguido passo a passo o processo de decodificação de Viterbi, que consiste em obter a seqüência de dados Z na saída do decodificador a partir da seqüência Y recebida do canal. Se a capacidade de correção do decodificador não for excedida, então espera-se que Z seja igual a V.

No decodificador convolucional de Viterbi, cada conjunto de dados que chega é um grupo de bits que equivale a uma saída de dados do codificador convolucional para cada k bits que entram no mesmo (para este exemplo, dois bits). A cada conjunto de bits recebidos deve-se medir a distância de Hamming entre eles e o valor esperado da saída do codificador convolucional para cada uma das possíveis transições a partir de cada um dos possíveis estados. Como neste exemplo a codificação convolucional é iniciada a partir do estado “00”, o primeiro conjunto de dados recebido deve ter a distância medida em relação à transição do estado “00” para ele mesmo e do estado “00” para o estado “10”.

Uma vez que o primeiro conjunto de dados, recebido no receptor digital, pode representar a transição a partir do estado “00” ao estado “00” ou ao estado “10”, o segundo conjunto de dados já pode representar as possíveis transições entre o estado “00” e ele mesmo, entre “00” e “10”, ou entre “10” e “01”, ou ainda entre “10” e “11”. Desta mesma forma deve-se medir a distância de todos os conjuntos de dados recebidos e todas as transições possíveis de serem representadas por eles ao longo do tempo.

A partir das primeiras transições, onde cada nó da treliça recebe apenas uma transição (ou seta, como as transições são representadas no diagrama de treliça), cada nó

passará a receber mais de uma transição (duas no caso do exemplo que está sendo apresentado). Cada uma destas transições terá uma distância de Hamming associada, obtida a partir da medida em relação ao dado de saída do codificador que esta transição representa e o dado recebido no decodificador. A cada transição deve ser associada uma métrica acumulada, que é a soma da sua distância de Hamming, em relação ao dado recebido, com a métrica acumulada do caminho sobrevivente que "chega" ao estado de origem da referida transição. Os caminhos são formados pelas seqüências de possíveis transições entre estados, seqüências estas que se formam ao longo do tempo. O caminho sobrevivente é um dos possíveis caminhos que chegam a um mesmo estado em um determinado instante de tempo e que contém a menor métrica acumulada. Para o caso deste exemplo, depois de iniciada a treliça, há sempre dois caminhos "chegando" a um mesmo estado em um mesmo instante de tempo. A técnica de Viterbi consiste em "matar" os caminhos menos prováveis que chegam a cada nó da treliça, deixando "vivos" aqueles com maior probabilidade de ocorrência. Portanto, no exemplo em questão, elimina-se o caminho de maior métrica acumulada, sobrevivendo o de menor.

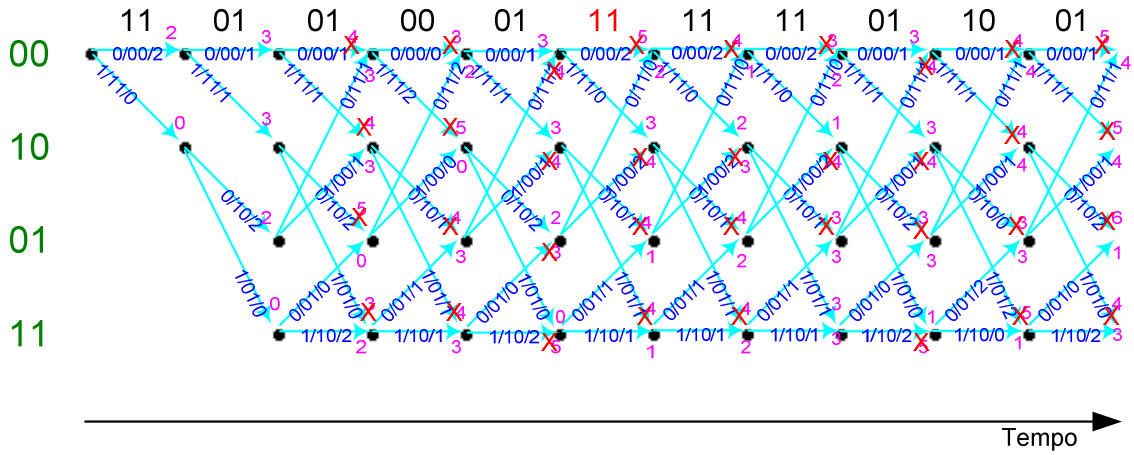

Figura 2.19: Diagrama de treliça com a representação da seqüência de dados recebidos (*yy* em preto) no decodificador convolucional do receptor digital. Representados, também, (em azul escuro) junto às transições no formato *v/xx/d*, o bit de entrada do codificador (*v*), os dados gerados na saída do codificador (*xx*) e a distância de Hamming (*d*) destes dados em relação aos dados recebidos *yy*. Em cor magenta, estão representados os valores das métricas acumuladas de cada um dos caminhos que chegam a cada nó da treliça, sendo sobrepostos por um "×" os caminhos eliminados.

Na Figura 2.19 está representada esta primeira parte das etapas da decodificação de Viterbi, descritas até agora neste exemplo. Nesta figura temos uma representação de um diagrama de treliça, onde para cada transição há um valor associado, em azul escuro, no formato *v/xx/d*, onde *v* representa o bit da seqüência *V* que entrou no codificador para

gerar tal transição, *xx* é o par de bits da seqüência X na saída do codificador e *d* é a distância de Hamming entre *xx* e o par de bits *yy* recebido a cada instante *n* através da seqüência Y. Cada instante *n* define uma célula da treliça. Neste exemplo, portanto, cada instante *n* implica na recepção de dois bits *yy* provenientes do canal. Acima de cada célula da treliça temos um respectivo valor *yy* em preto, exceto o valor em vermelho acima da célula que representa o instante *n* em que os dados foram corrompidos pela ação degradante do canal. Nesta mesma Figura 2.19, um "×" vermelho representa o caminho eliminado na chegada a cada estado, por apresentar transições com maior métrica acumulada, e um valor em magenta indicando a métrica acumulada de cada caminho.

Ainda na Figura 2.19, é possível observar a existência de onze instantes de tempo *n* (células) em que estão representados os dados que chegam ao receptor e todo o conjunto de transições a serem analisadas. Esta quantidade limitada deve-se ao fato de ter-se assumido, para este exemplo, que o número de instantes de tempo *n* de funcionamento do codificador convolucional antes que este seja reiniciado é onze. A este número de instantes de tempo *n* entre reinicializações, chamaremos de profundidade do decodificador.

Uma vez recebida a quantidade de dados igual à profundidade do decodificador, começa-se o processo de *traceback*, que consiste em seguir de traz para frente, na treliça, as transições que formam o caminho de menor métrica acumulada. Este caminho de menor métrica acumulada, no exemplo aqui apresentado, está representado na Figura 2.20, onde todos os outros caminhos ilustrados na Figura 2.19 tiveram sua representação suprimida.

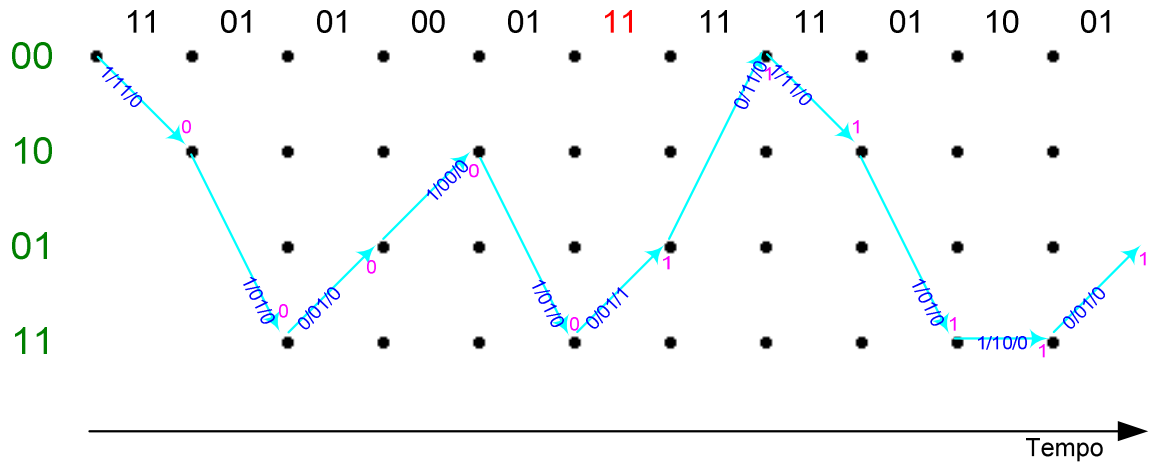

Figura 2.20: Representação apenas do caminho de menor métrica acumulada sobre o diagrama de treliça, pronto para a execução da rotina de ***traceback***, visando a obtenção da seqüência de dados com maior chance de equivaler aos dados codificados no codificador convolucional do transmissor digital.

Utilizando-se da técnica do *traceback*, pode-se então obter a estimativa da seqüência de valores do bit *v* na entrada do codificador, isto é, pode-se então obter a seqüência de dados decodificados Z. O *traceback* consiste em registrar o valor *v* mostrado no *label v/xx/d* em cada transição representada na Figura 2.20, resultando em:

Z: 1 1 0 1 1 0 0 1 1 1 0

Note neste exemplo que, mesmo tendo ocorrido um erro nos dados recebidos, houve êxito na correção deste erro na decodificação, pois os dados V originalmente codificados no transmissor digital foram recuperados integralmente no receptor digital. O sucesso do processo de correção de erro é atestado pelo fato de $Z = V$.

Algo importante a ser levado em conta é observável na Figura 2.19, onde, em determinados estados da treliça, "chegam" mais de uma transição pertencentes a caminhos contendo a mesma métrica acumulada. A política adotada nestes casos foi a escolha aleatória de um dos dois caminhos a ser eliminado. Esta escolha aleatória não interferiu no resultado da decodificação por ter ocorrido em caminhos com métricas superiores a do caminho dos dados mais próximos aos codificados no codificador convolucional do transmissor digital. Porém quanto maior o número de erros inseridos na seqüência recebida, maior a probabilidade de haver mais caminhos com métricas iguais, inclusive sendo possível eliminar o caminho que represente o conjunto de transições que foram geradas no codificador convolucional. Neste caso, diz-se que houve um estouro na capacidade de correção do decodificador convolucional de Viterbi, e, portanto, os dados resultantes da codificação podem nada ter a ver com aqueles que foram originalmente codificados no transmissor digital. A melhor maneira de se evitar este tipo de ocorrência é escolhendo-se códigos convolucionais com as maiores distâncias livres possíveis, o que garante maiores distâncias de Hamming entre os demais caminhos e o caminho que representa a correta seqüência de transições originadas na codificação convolucional no transmissor digital.

A profundidade do decodificador, do exemplo das Figuras 2.19 e 2.20, foi determinada como sendo onze apenas para que o gráfico de treliça ficasse mais fácil de ser representado e visualizado. Na verdade a profundidade de um decodificador poderia ser toda a seqüência de dados transmitida entre o receptor e o transmissor. Esta abordagem de uma seqüência muito extensa de dados tem algumas incompatibilidades com a prática: (1) Se a seqüência de dados a ser decodificada for muito grande levará grandes espaços de tempo para que se possa fazer o *traceback*. Se a seqüência de dados for de tamanho

infinito, jamais será possível decodificar os dados. (2) Da mesma forma, quanto maior a profundidade, maior a quantidade de memória necessária para armazenamento das informações sobre as transições ao longo da treliça.

Pela dificuldade em decodificar-se seqüências de dados de grandes extensões, determina-se uma dimensão fixa para o receptor. Toda a vez que a profundidade da memória do receptor é atingida, ao registrarem-se os dados que vão chegando ao receptor digital, faz-se o *traceback* e obtém-se uma parte dos dados decodificados. Geralmente os primeiros dados que chegam formam, na treliça do decodificador de Viterbi, caminhos de transições com métricas acumuladas que não levam necessariamente ao conjunto de dados originalmente codificados no transmissor digital. Porém, ao aumentarem-se as extensões dos caminhos com a chegada de novos dados, as métricas acumuladas vão convergindo de forma a indicarem o caminho que melhor represente os dados transmitidos. A literatura [1] indica como sendo de no mínimo $5m$ a profundidade necessária para garantir uma perda desprezível no desempenho do decodificador de Viterbi de taxa de codificação de 1/2, se comparado frente a um decodificador teórico de profundidade infinita.

Há duas diferentes abordagens sobre como iniciar e reiniciar (após cada vez que a profundidade do decodificador é atingida) o preenchimento da memória do decodificador convolucional de Viterbi: (1) O modo de operação contínuo é a abordagem em que o codificador convolucional no transmissor codifica contínua e ininterruptamente os dados em sua entrada. Para esta abordagem o decodificador pode ser reiniciado ou a partir do estado que corresponda à continuidade do caminho de menor métrica do ciclo de *trace back* anterior, ou sempre de um mesmo estado pré-determinado (geralmente o estado zero). (2) O modo de operação truncado exige que haja um sincronismo no funcionamento do codificador e do decodificador convolucionais. Nesta abordagem, o codificador convolucional é reiniciado, e levado a um estado conhecido e pré-determinado, toda a vez que este tiver codificado um número de dados igual à profundidade do decodificador. Por sua vez, o decodificador volta sempre ao mesmo estado (o mesmo estado inicial do codificador) após cada ciclo de *traceback*. Com o CODEC (conjunto de codificador e decodificador) convolucional funcionando em modo truncado, reduzem-se as chances das métricas acumuladas, nas seqüências formadas pelas primeiras transições dos caminhos na treliça no decodificador convolucional de Viterbi, não corresponderem ao caminho que represente a correta seqüência de dados codificada pelo codificador convolucional, aumentando-se, assim, ligeiramente o desempenho da correção de erros.

2.4 *Soft-Decision*, *Soft*-Bits e o *demapper* com *soft-output*

Conforme visto no capítulo anterior, os dados resultantes na saída da etapa de codificação convolucional no transmissor digital são representados pela seqüência de bits X e os dados recebidos no receptor digital, após passagem pelo canal, são representados pela seqüência de bits Y. Tanto mais dessemelhantes entre si serão as seqüências X e Y, quanto maior for o ruído e/ou interferência impostos pelo canal. A dessemelhança entre X e Y é medida pelo número de bits com valor lógico distinto entre estas seqüências.

Esta maneira de avaliar comparativamente a semelhança entre X e Y é conhecida como *Hard-Decision*, porque a decisão sobre a semelhança é quantizada de forma binária, isto é, não existe um meio termo – ou um determinado bit de X é igual ao seu correspondente em Y, ou então não é igual. Alternativamente a esta abordagem, será apresentado neste capítulo o método *Soft-Decision* de tratamento dos dados, método este que é o foco principal deste trabalho. Será utilizado como base para introdução deste método o texto apresentado em [3].

Conforme já discutido na introdução deste Capítulo 2, cada palavra binária, quando transmitida pelo transmissor digital, é transportada por um sinal de RF denominado de portadora. Especificamente, cada símbolo $s = I + jQ$ gerado no modulador (*mapper*) a partir de cada palavra binária a ser transmitida, é tal que o módulo de s representa a amplitude da portadora e o ângulo de s representa a fase da portadora. O tempo T durante o qual a portadora mantém uma determinada amplitude e uma determinada fase corresponde ao intervalo de duração do símbolo, que, por sua vez, corresponde ao intervalo de duração da palavra binária mapeada no símbolo. O parâmetro $1/T$ é denominado de *symbol rate*, e, como o nome sugere, expressa o número de símbolos transmitidos por unidade de tempo.

O conjunto de possíveis amplitudes e fases é determinado pelo tipo de modulação adotado (QAM, PSK, etc...), e, portanto, implica em valores de amplitude e fase quantizados de forma a representar cada um dos possíveis símbolos da constelação

transmitidos a cada instante T. Desta maneira, estes valores quantizados representam um número determinado de possíveis palavras binárias que podem ser transmitidas. Estes valores, pertencentes ao conjunto de valores discretos que representam as possíveis amplitudes e fases da portadora, são, portanto, valores que identificam os símbolos da constelação de uma maneira *hard* – ou o valor quantizado de amplitude e fase corresponde a um determinado símbolo ou não.

Quando uma portadora é transmitida através de um canal AWGN (Additive White Gaussian Noise) ocorre a soma da mesma a um outro sinal de origem aleatória com uma distribuição gaussiana de média zero. Sejam os símbolos $s_1(T) = I_1 + jQ_1$ e $s_2(T) = I_2 + jQ_2$, cada um deles transmitidos em um intervalo de um símbolo $(0,T)$. Sejam $s_1(T)$ e $s_2(T)$ gerados na saída do *mapper* para respectivamente as palavras binárias 1 e 0 em sua entrada. O sinal recebido, no receptor digital, será dado por $r(t) = s_i(t) + n(t)$, $i \in \{1,2\}$, onde $n(t)$ é o ruído aditivo gaussiano de média zero imposto pelo canal. No receptor digital o sinal recebido é tratado como uma variável $z(T) = s_i + n_0$, onde s_i e n_0 são respectivamente a componente do sinal transmitido pelo transmissor digital e a componente do ruído aditivo gaussiano imposto pelo canal AWGN. A componente n_0 é uma variável aleatória gaussiana de média zero. Portanto $z(T)$ é uma variável aleatória gaussiana de média s_1 ou s_2, dependendo do dado binário que foi enviado pelo transmissor digital ter sido de valor 1 ou valor 0.

A partir da recepção do sinal analógico $z(T)$ na entrada do receptor digital, é necessário executar um processo de decisão em que se determine a qual sinal, dos possíveis de serem transmitidos pelo transmissor digital, o sinal recebido corresponde. Esta decisão é tomada comparando-se o sinal $z(T)$ a um *threshold* (limiar, da representação linear, entre dois valores discretos, geralmente o ponto médio entre eles). As probabilidades condicionais de $z(T)$, $p(z \mid s_1)$ e $p(z \mid s_2)$ estão representadas na Figura 2.21, como características de um sistema hipotético e genérico, usado como exemplo.

O *Hard-Decision* é o modo de funcionamento do decodificador convolucional para quando os bits dos dados em sua entrada são representados apenas por dois níveis possíveis. Neste caso diz-se que o sinal entregue a ele pelo *demapper* tem apenas dois níveis de quantização.

O *damapper* pode, ainda, fornecer dados, ao decodificador convolucional, com mais de dois níveis de quantização, caso em que o decodificador funciona no chamado modo *Soft-Decision*. No exemplo ilustrado na Figura 2.21 estão representados, sob o eixo das abscissas, os intervalos de tomada de decisão para o funcionamento do decodificador no modo *Soft-Decision* com oito níveis de quantização. Neste caso, ao contrario do caso do *Hard-Decision*, os bits entregues ao decodificador são representados por uma palavra binária de mais de um bit (três bits neste exemplo), o que permite ao decodificador convolucional ter uma informação a mais, que corresponde ao "grau de confiabilidade" da decisão tomada pelo *demapper*. Pode-se dizer que, no exemplo da Figura 2.21, uma palavra binária de valor 111, gerada pelo *demapper*, pode ser "considerada" pelo decodificador como sendo um bit de valor 1 com um altíssimo grau de certeza, enquanto no caso do recebimento de um valor 100, pode-se considerar como um bit de valor 1 com um grau de certeza muito baixo.

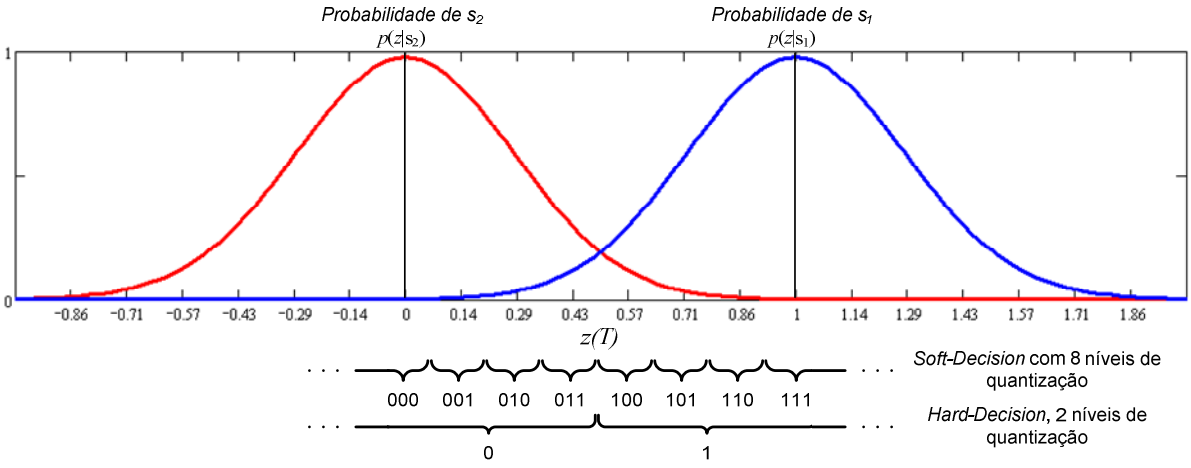

Figura 2.21: Incerteza de ocorrência do valor 0 ou 1 na recepção de um sinal digital binário em um receptor digital que recebe um sinal gerado em um transmissor digital. Em vermelho é representada a probabilidade de um sinal transmitido com o valor 0 ser interpretado como sendo 0 e em azul a probabilidade de um sinal transmitido como 1 ser interpretado como 1, no receptor digital após passagem pelo canal aditivo gaussiano. Sob as curvas do gráfico estão representados dois conjuntos de possíveis intervalos usados para a tomada de decisão a respeito do valor transmitido pelo transmissor digital ao qual o sinal recebido corresponda, para *Soft* e *Hard-Decision*.

Para que os bits na saída do decodificador convolucional de Viterbi no receptor digital possam ser processados ao longo das etapas seguintes do sistema de recepção digital, é de vital importância que eles estejam em um formato *hard*, em que um bit da seqüência original de dados, codificado e transmitidos pelo transmissor digital, seja representado por um único bit. O aumento na quantidade de informação contida em um bit, representado por uma palavra binária de mais de um bit, na comunicação entre o *demapper* e um decodificador convolucional de Viterbi funcionando em modo *Soft-Decision*, tem

como único objetivo auxiliar, aumentando a precisão, na tarefa de correção de erros executada pelo decodificador.

Para correção de erros, causados em dados digitais transmitidos em um canal AWGN, a utilização de oito níveis de quantização (três bits) permite que o decodificador convolucional de Viterbi, funcionando em modo *Soft-Decision* e com taxa de codificação de 1/2, corrija a mesma quantidade de erros que um codificador convolucional de Viterbi, funcionando em modo *Hard-Decision* com mesma taxa, corrigiria da mesma seqüência de dados, se transmitidos em um canal aditivo gaussiano com uma SNR (relação sinal-ruído) 3dB mais alta [3]. Se fossem utilizados infinitos níveis de quantização para representar os dados na entrada de um decodificador de Viterbi, funcionando em modo *Soft-Decision*, o desempenho deste decodificador seria equivalente ao de um decodificador de Viterbi, funcinando em modo *Hard-Decision*, que tivesse de corrigir dados vindos de um canal com uma SNR de até 3.5dB maior [1][2]. Quanto maior o nível de quantização dos dados processados em um decodificador convolucional de Viterbi, maior o custo computacional e o uso de memória para armazenamento dos dados necessários à execução dos algoritmos pertinentes.

Para que um sistema baseado em tecnologia digital, como o caso de um receptor de TV digital, tenha um baixo custo juntamente com um grande desempenho, há a necessidade de utilizar-se a menor quantidade possível de memória, bem como representar-se os dados com o menor número possível de bits. Buscando-se este compromisso entre custo e desempenho, considera-se que a melhor configuração, para um decodificador convolucional de Viterbi, é o funcionamento em modo *Soft-Decision* utilizando-se de oito níveis de quantização (três bits).

Chama-se de *demapper* com *soft output* o *demapper* que tem em sua saída símbolos que contenham bits representados por mais de dois níveis de quantização (um *demapper* compatível com a decodificação convolucional com *Soft-Decision*). Os bits obtidos a partir da quantização com mais de dois níveis no *demapper* e representados por palavras binárias de mais de um bit, são chamados de *soft*-bits.

3 Implementação

A descrição da implementação de um decodificador convolucional de Viterbi com *Soft-Decision* para sistemas de televisão digital terrestre que utilizem modulação M-QAM e bit *interleaver*, é o objetivo principal deste trabalho. Neste capítulo será descrita a heurística de implementação, em linguagem de programação ANSI C, do CODEC (conjunto de codificador e decodificador) convolucional. Foi implementado o CODEC para os sistemas de transmissão e recepção de televisão digital terrestre, europeu (DVB-T) e japonês (ISDB-T).

Ao longo deste capítulo serão descritos cada um dos principais passos que foram necessários à implementação do CODEC convolucional, a que este trabalho se refere. O funcionamento, teste e validação deste simulador dependem da interação com outros blocos específicos para modulação digital, desenvolvidos pelos demais pesquisadores do CPTW. Tais blocos terão suas funções brevemente descritas, assim como a forma com que interagem com os blocos desenvolvidos para o CODEC convolucional. Não serão descritos os detalhes do funcionamento da forma de implementação e nem sobre a validação destes blocos por este assunto fugir do escopo deste trabalho.

As normas para os sistemas DVB-T e ISDB-T [18][19] estabelecem a utilização de modulação M-QAM para o mapeamento dos símbolos que representarão os dados a serem transmitidos. Esta modulação é, ainda, passível de ser configurada para a utilização de constelações de diferentes densidades. Uma vez que os dois sistemas têm em comum a possibilidade de mapear seus símbolos a serem transmitidos em constelações QPSK, 16-QAM e 64-QAM, decidiu-se que o simuladores seriam implementados com a possibilidade de configurar-se a utilização de qualquer uma das três citas modulações. O mapeamento *Gray* correspondente às normas dos sistemas aqui citados para a modulação 16-QAM, está representado na constelação da Figura 2.3.

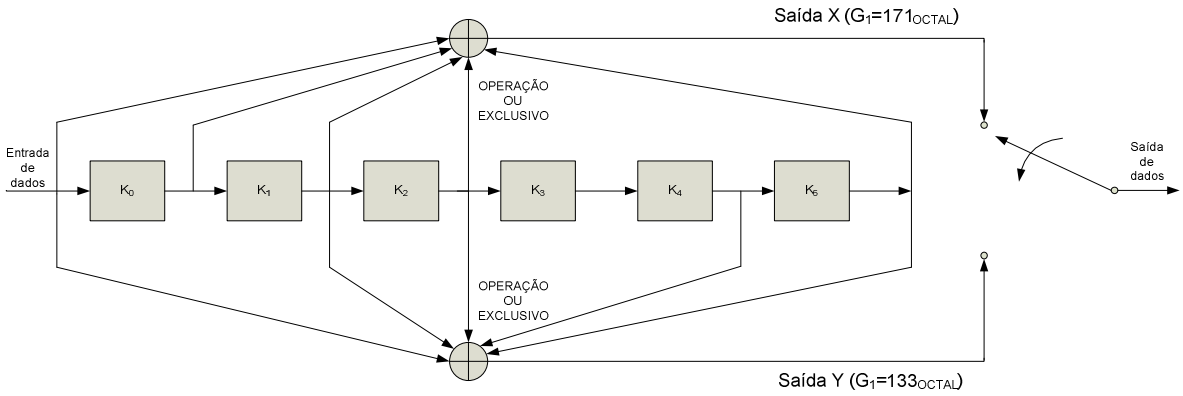

Figura 3.1: Diagrama de blocos do codificador convolucional com taxa de codificação de ½ dos sistemas DVB-T e ISDB-T para transmissão de TV digital terrestre.

O codificador convolucional, adotado por ambos os sistemas, tem sua representação, em diagrama de blocos, ilustrada na Figura 3.1. Os polinômios geradores deste codificador, que tem uma taxa de codificação de 1/2, em formato octal, são dados por $G_1 = 171_{OCTAL}$ e $G_2 = 133_{OCTAL}$.

O codificador da Figura 3.1 pode ter sua taxa de codificação aumentada para taxas de 2/3, 3/4, 5/6 e 7/8, aplicando-se as matrizes de *puncturing*, conforme Tabela 2, prevista em ambas as normas.

Tabela 1: Matrizes de *puncturing* para diferentes taxas de codificação, aplicáveis à saída do codificador convolucional apresentado na Figura 3.1.

Taxa de codificação	Matrizes de *puncturing*	Saída do codificador (depois do *puncturing* e da serialização dos dados)
1/2	X: 1 Y: 1	$X_1\ Y_1$
2/3	X: 1 0 Y: 1 1	$X_1\ Y_1\ Y_2$
3/4	X: 1 0 1 Y: 1 1 0	$X_1\ Y_1\ Y_2\ X_3$
5/6	X: 1 0 1 0 1 Y: 1 1 0 1 0	$X_1\ Y_1\ Y_2\ X_3\ Y_4\ X_5$
7/8	X: 1 0 0 0 1 0 1 Y: 1 1 1 1 0 1 0	$X_1\ Y_1\ Y_2\ Y_3\ Y_4\ X_5\ Y_6\ X_7$

3.1 Codificador Convolucional

A Figura 3.2 mostra o fluxograma do algoritmo que executa as funções do codificador convolucional desenvolvido neste trabalho.

Para implementação do *shift register* do codificador convolucional, em software, utilizou-se uma memória de sete posições, diferindo da representação em diagrama de blocos apresentada na Figura 3.1, em que este registrador possui apenas seis posições. Diferentemente de um sistema implantado em *hardware*, em que há um fluxo de dados constante, na abordagem em *software* é preciso que todos os dados necessários à execução de uma operação, lógica ou matemática, estejam previamente disponíveis para que possam ser "enviados" à unidade de processamento. Ao se observar a Figura 3.1, nota-se que, no caso de executarem-se as operações lógicas para a obtenção dos valores de saída "X" e "Y" em um período de "*clock*" t, esta operação estará baseada em dados contidos nas posições do *shift register* no instante de tempo $t-1$ **e** no dado que está entrando no sistema, também neste mesmo instante anterior à operação. Por esta razão foi acrescentada uma posição de memória a mais na máquina de estados, visando "segurar" o dado da entrada do codificador convolucional até que todas as operações estejam concluídas.

No fluxograma da Figura 3.2, onde se lê "desloca os dados para a direita", está sendo feita uma referência à topologia do *shift register* representado na Figura 3.1. Nesta operação, entra um novo bit na primeira posição da memória do *shift register* e último bit é descartado.

A profundidade do codificador, utilizada para as simulações, foi obtida através de testes de hipótese até encontrar-se o valor 136. A utilização deste valor propiciou o melhor desempenho obtido em simulações com constelações 16 e 64-QAM.

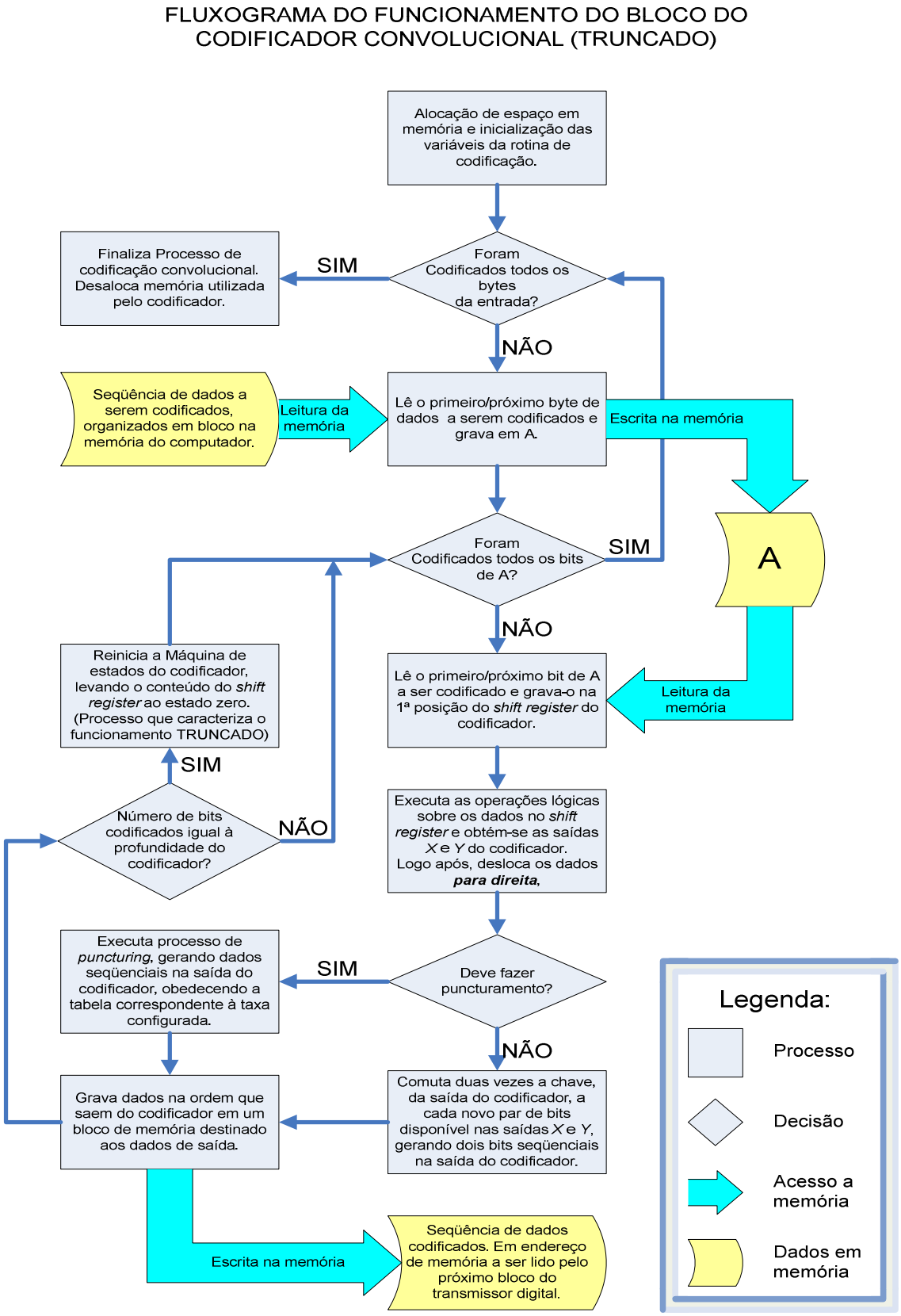

Figura 3.2: Fluxograma do algoritmo do codificador convolucional do simulador do transmissor digital de TV (de acordo com os sistemas DVB-T e ISDB-T).

3.2 *Mapper* e *demapper*

A principal motivação para o desenvolvimento da pesquisa que deu origem à escrita deste trabalho, originou-se na não obviedade da forma de implementar-se um decodificador convolucional de Viterbi com *Soft-Decision* em receptores digitais que façam parte de sistemas de comunicação digital que utilizem bit *interleaver* e modulações M-QAM. Por esta razão este trabalho focaliza a implementação de um simulador que apenas simula situações em que possam ser utilizadas modulações 16-QAM ou 64-QAM. Os sistemas ISDB-T e DVB-T, de transmissão de TV digital, aos quais se buscou simular em ambiente computacional, também contemplam a possibilidade de modulações QPSK e DQPSK, deixadas de fora deste trabalho.

3.2.1 *Mapper*

A implementação da etapa de mapeamento em símbolos IQ das palavras binárias geradas na etapa de codificação de canal de um sistema de comunicação digital M-QAM tem uma grande variedade de formas computacionalmente simples, possíveis de serem abordadas. A forma escolhida para ser usada na geração do simulador de *Mapper* foi o tabelamento, em memória, dos possíveis diferentes valores I e Q que deveriam ser gerados para cada uma das possíveis palavras binárias originadas na etapa de codificação de canal do transmissor digital. Este tabelamento seguiu a forma de mapear uma constelação 16-QAM conforme apresentada na Figura 2.3 e uma constelação 64-QAM conforme apresentada na Figura 3.3, que estão de acordo com os mapas apresentados nas normas dos sistemas DVB-T e ISDB-T de TV digital [18][19].

O mapeamento apresentado na Figura 3.3 segue o formato "I_2 Q_2 I_1 Q_1 I_0 Q_0", em que Q_0 representa o bit menos significativo da palavra de seis bits mapeada e I_2 representa o bit mais significativo desta palavra. A distribuição dos símbolos sobre o mapa da constelação da Figura 3.3 obedece ao código Gray, porém o modo como este código foi arranjado será objeto discutido em seguida na sessão sobre o processo de "demapeamento".

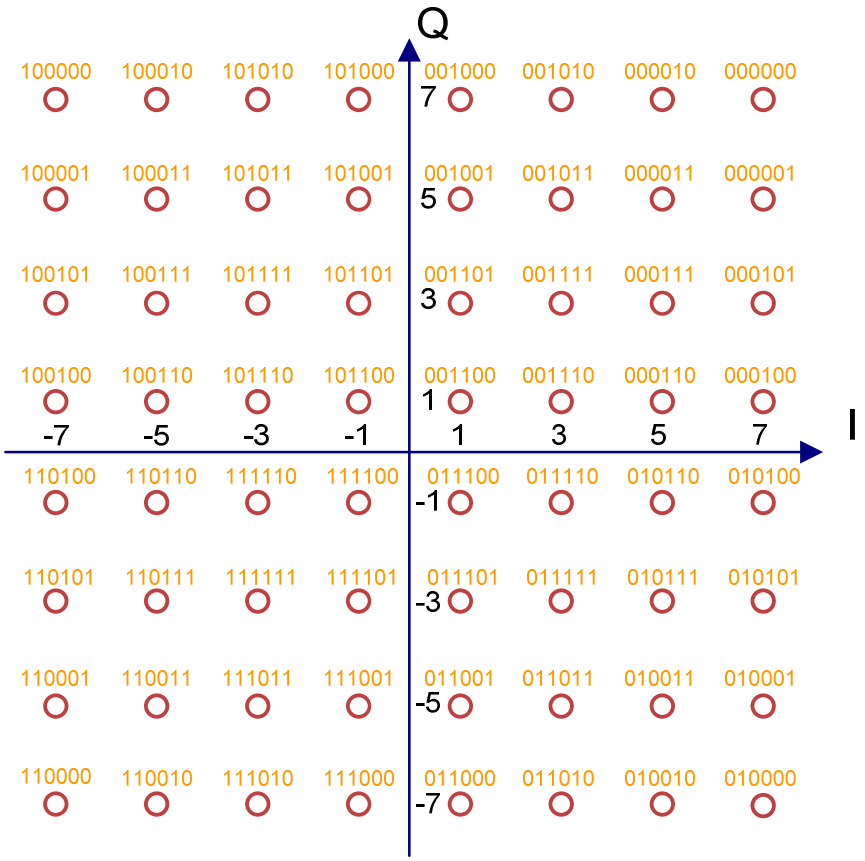

Figura 3.3: Representação gráfica da constelação resultante da modulação 64-QAM. Usada em sistemas de transmissão de TV digital, tais como DVB-T e ISDB-T.

3.2.2 *Demapper*

Para o projeto de um receptor digital de um sistema de comunicação digital com modulação M-QAM e intercalação de bits, em que se deseje que o decodificador convolucional de Viterbi funcione no modo *Soft-Decision*, é necessário que haja uma especial atenção no projeto do sistema de "demapeamento", conforme já citado anteriormente. A referida atenção diz respeito à necessidade de propagar-se a informação referente ao erro de posicionamento do símbolo recebido, em relação à constelação padrão, até o decodificador convolucional de Viterbi, que fará uso desta informação para gerar um aumento no desempenho de sua tarefa de correção de erros. Uma vez que em tal sistema os dados, após "demapeados", são desembaralhados pelo bit *deinterleaver*, não é possível determinar a qual símbolo um bit, na entrada do decodificador convolucional de Viterbi, pertencera durante a etapa de propagação através do canal de RF.

Em sistemas desta natureza, a solução mais apropriada para propagarem-se as informações sobre o erro de posicionamento dos símbolos recebidos no receptor digital, é a utilização de *soft*-bits que tragam em si a sua colaboração individual para o erro de posicionamento do símbolo que os "transportou" através do canal [5]. Nos sistemas de transmissão de sinal de televisão digital DVB-T e ISDB-T, focos deste estudo, utiliza-se um mapeamento dos símbolos da modulação que permite dividir-se a constelação de símbolos em áreas cujas interseções formam quadrantes e sub-quadrantes.

Nas Figuras 3.4 e 3.5, inspiradas nas figuras encontradas na publicação de Filippo Tosato [5], estão representadas as áreas utilizadas para gerar os *soft*-bits na saída do *demapper*.

O mapeamento 16-QAM representado nas imagens da Figura 3.4 é de formato "I_1 Q_1 I_0 Q_0" (com I_1 sendo o bit mais significativo e Q_0 o menos). Sobre cada um dos círculos vermelhos, que representam as posições de referência dos símbolos QAM nas constelações da Figura 3.4, há um valor binário correspondendo a cada um dos símbolos. Desta forma é possível observar que todos os símbolos da metade representada no interior de um quadro vermelho da Figura 3.4a têm o valor "1" em seus bits I_1 e o valor zero nos mesmos bits dos símbolos na metade oposta (quadro azul). Isto significa que o bit I_1, desta configuração, determina em qual das duas metades, direita ou esquerda, encontra-se o símbolo que representa a palavra binária a qual ele pertence. E ainda, o bit I_1 representa o sinal do valor do símbolo sobre o eixo I (eixo dos valores reais). Da mesma forma, o bit Q_1 determina se o símbolo recebido está localizado na metade superior ou inferior do plano IQ e representa o sinal do valor do símbolo sobre o eixo Q que representa a palavra binária a qual ele pertence, conforme se pode observar na Figura 3.4b.

Valendo-se do mesmo princípio, utiliza-se o bit I_0 para determinar-se a qual das metades, da metade da escala de I já determinada por I_1, pertence um suposto símbolo recebido no receptor digital. Para isto é necessário deslocar-se o centro da área de decisão para entre as duas possíveis posições de referência às quais o símbolo possa corresponder. A variação possível fica entre mais um e menos um em relação ao centro das posições de referência, sendo novamente a metade positiva representada pelo valor binário zero e a metade negativa pelo valor um (Figuras 3.4c e 3.4d). Para a execução deste deslocamento basta (para o caso de uma constelação 16-QAM, como as da Figura 3.4) subtrair dois do valor absoluto da posição do símbolo recebido.

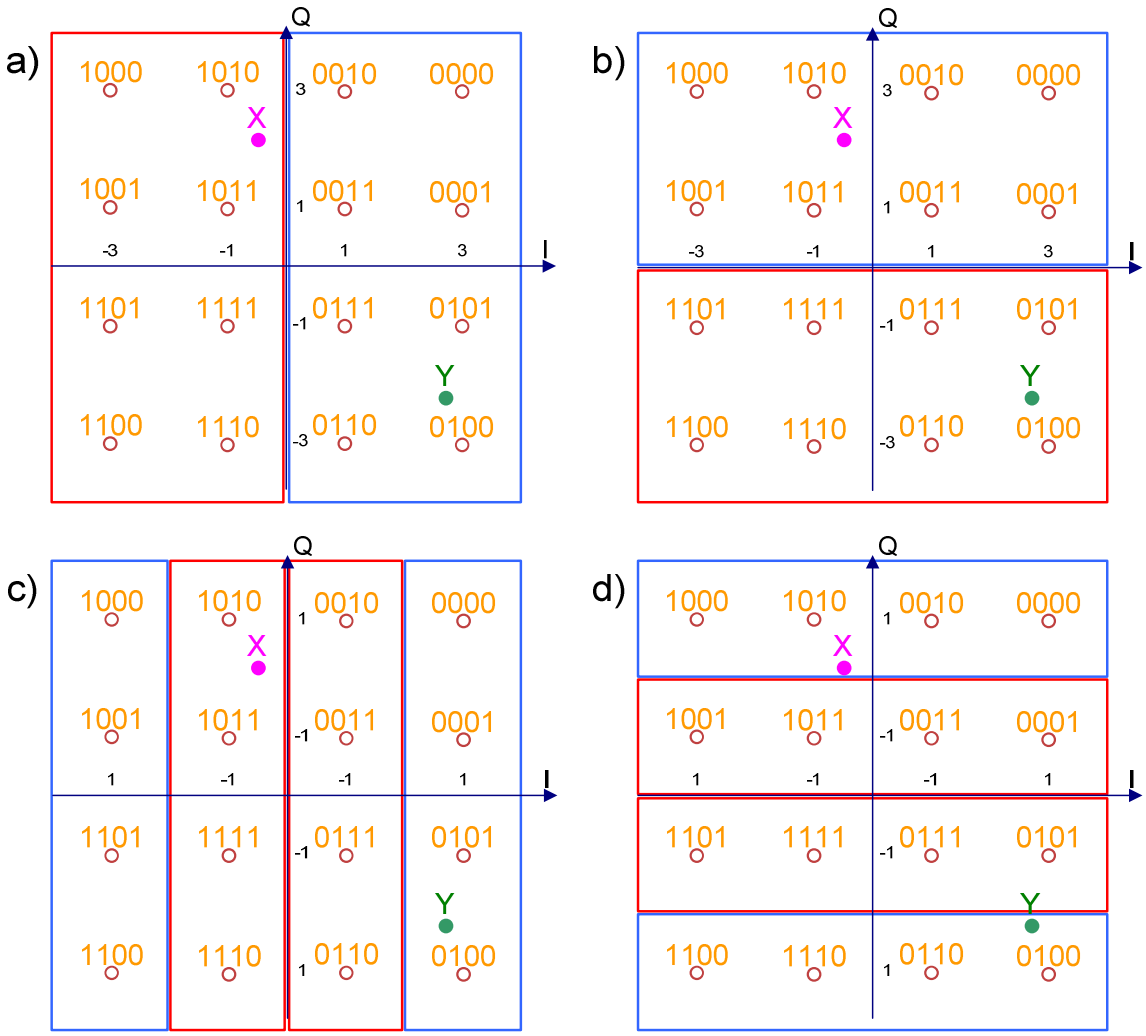

Figura 3.4: Representação gráfica de constelações 16-QAM, sobrepostas pelas áreas utilizadas para a geração de *soft*-bits pelo *demapper*.

Para melhor ilustrar a descrição feita nos dois parágrafos anteriores será apresentado um exemplo. Neste exemplo consideraram-se dois símbolos hipotéticos X e Y, representados por pontos de cor magenta e verde, respectivamente, nos mapas representados na Figura 3.4.

Para obterem-se os *hard*-bits I_{Xh1}, Q_{Xh1}, I_{Xh0}, Q_{Xh0}, I_{Yh1}, Q_{Yh1}, I_{Yh0} e Q_{Yh0} utilizando-se do método *Hard-Decision*, deve-se simplesmente considerar "linhas de decisão" que fiquem exatamente na metade da distância entre as posições dos símbolos de referência e determinar de qual destas posições o símbolo recebido está mais próximo. Portanto deve-se detectar se o símbolo recebido está de um lado ou de outro da linha imaginária entre duas posições de referência da constelação.

A obtenção dos *soft*-bits I_{XS1}, Q_{XS1}, I_{XS0}, Q_{XS0}, I_{YS1}, Q_{YS1}, I_{YS0} e Q_{YS0} é feita inicialmente atribuindo-se aos *soft*-bits I_{XS1}, Q_{XS1}, I_{YS1} e Q_{YS1} as coordenadas I e Q dos símbolos recebidos (valores que, como será visto posteriormente, deverão ser quantizados). Portanto, conforme a Figura 3.4:

$$I_{XS1} \cong -0{,}45$$

$$Q_{XS1} \cong 2{,}2$$

$$I_{YS1} \cong 2{,}7$$

$$Q_{YS1} \cong -2{,}7$$

Para obterem-se os *soft*-bits I_{XS0}, Q_{XS0}, I_{YS0} e Q_{YS0}, conforme citado anteriormente, deve-se subtrair dois dos valores absolutos dos *soft*-bits I_{XS1}, Q_{XS1}, I_{YS1} e Q_{YS1}. Esta operação permite decidir o grau de colaboração dos bits I_{XS0}, Q_{XS0}, I_{YS0} e Q_{YS0} para a determinação de a qual das metades das áreas, já restringidas pelos bits I_{XS1}, Q_{XS1}, I_{YS1} e Q_{YS1}, os símbolos recebidos pertencem. Assim sendo:

$$I_{XS0} = |I_{XS1}| - 2 = 0{,}45 - 2 = -1{,}55$$

$$Q_{XS0} = |Q_{XS1}| - 2 = 2{,}2 - 2 = 0{,}2$$

$$I_{YS0} = |I_{YS1}| - 2 = 2{,}7 - 2 = 0{,}7$$

$$Q_{YS0} = |Q_{YS1}| - 2 = 2{,}7 - 2 = 0{,}7$$

Considera-se que a variação possível para os *soft*-bits I_{XS1}, Q_{XS1}, I_{YS1} e Q_{YS1} seja de menos três até três (Figuras 3.4a e 3.4b), portanto diz-se que um destes bits tem maior chance de ter seu valor "um" quanto mais próximo de menos três for o valor de seu *soft*-bit correspondente e maior chance de ter seu valor "zero", quanto mais próximo de três. Para os *soft*-bits I_{XS0}, Q_{XS0}, I_{YS0} e Q_{YS0}, consideram-se a variação entre menos um e um, estando o valor menos um do *soft*-bit vinculado à maior certeza de o bit correspondente valer um e o valor um à maior certeza de o bit correspondente valer zero. No caso do *Soft-Decision* a decisão final quanto ao valor binário do bit fica delegada à etapa de decodificação de Viterbi.

Para apresentar a regra de obtenção de *soft*-bits de maneira mais formal, consideremos um símbolo IQ de uma constelação 16-QAM representado por *soft*-bits I_{S1}, Q_{S1}, I_{S0}, Q_{S0} e com coordenadas (I_r,Q_r) no plano IQ. Tem-se então:

$$I_{S1} = I_r$$

$$Q_{S1} = Q_r$$

$$I_{S0} = |I_{S1}| - 2$$

$$Q_{S0} = |Q_{S1}| - 2$$

Esta regra pode ser extrapolada para qualquer densidade de constelação M-QAM formada pelo código Gray compatível. Na Figura 3.5 estão representadas as áreas de decisão para a obtenção de *soft*-bits em uma constelação 64-QAM. Os valores atribuídos aos seis *soft*-bits I_{S2}, Q_{S2}, I_{S1}, Q_{S1}, I_{S0}, Q_{S0}, os quais representam um símbolo da constelação 64-QAM, são obtidos de forma similar à usada para obtenção dos *soft*-bits para uma constelação 16-QAM. Desta forma os valores dos *soft*-bits I_{S2} e Q_{S2} podem variar entre menos sete e mais sete (Figuras 3.5a e 3.5b), dos *soft*-bits I_{S1} e Q_{S1} podem variar entre menos três e mais três (Figuras 3.5c e 3.5d) e dos *soft*-bits I_{S0} e Q_{S0}, entre menos um e mais um (Figuras 3.5e e 3.5f). A seguir a regra para obterem-se os valores dos *soft*-bits I_{S2}, Q_{S2}, I_{S1}, Q_{S1}, I_{S0}, Q_{S0} referentes a um símbolo IQ de uma constelação 64-QAM com coordenadas (Ir,Qr) no plano IQ:

$$I_{S2} = I_r$$

$$Q_{S2} = Q_r$$

$$I_{S1} = |I_{S2}| - 4$$

$$Q_{S1} = |Q_{S2}| - 4$$

$$I_{S0} = |I_{S1}| - 2$$

$$Q_{S0} = |Q_{S1}| - 2$$

Conforme visto no exemplo para a obtenção dos *soft*-bits correspondentes aos símbolos X e Y representados nas imagens da Figura 3.4, os valores atribuídos a cada um dos bits são de característica linear e poderiam ser representados, em um algoritmo para sistema digital, por variáveis de ponto flutuante. Contudo, conforme explicado no capítulo 2.4, quanto menor a quantidade de bits utilizada para representar os *soft*-bits, mais barato e eficiente será o sistema digital de decodificação destes dados.

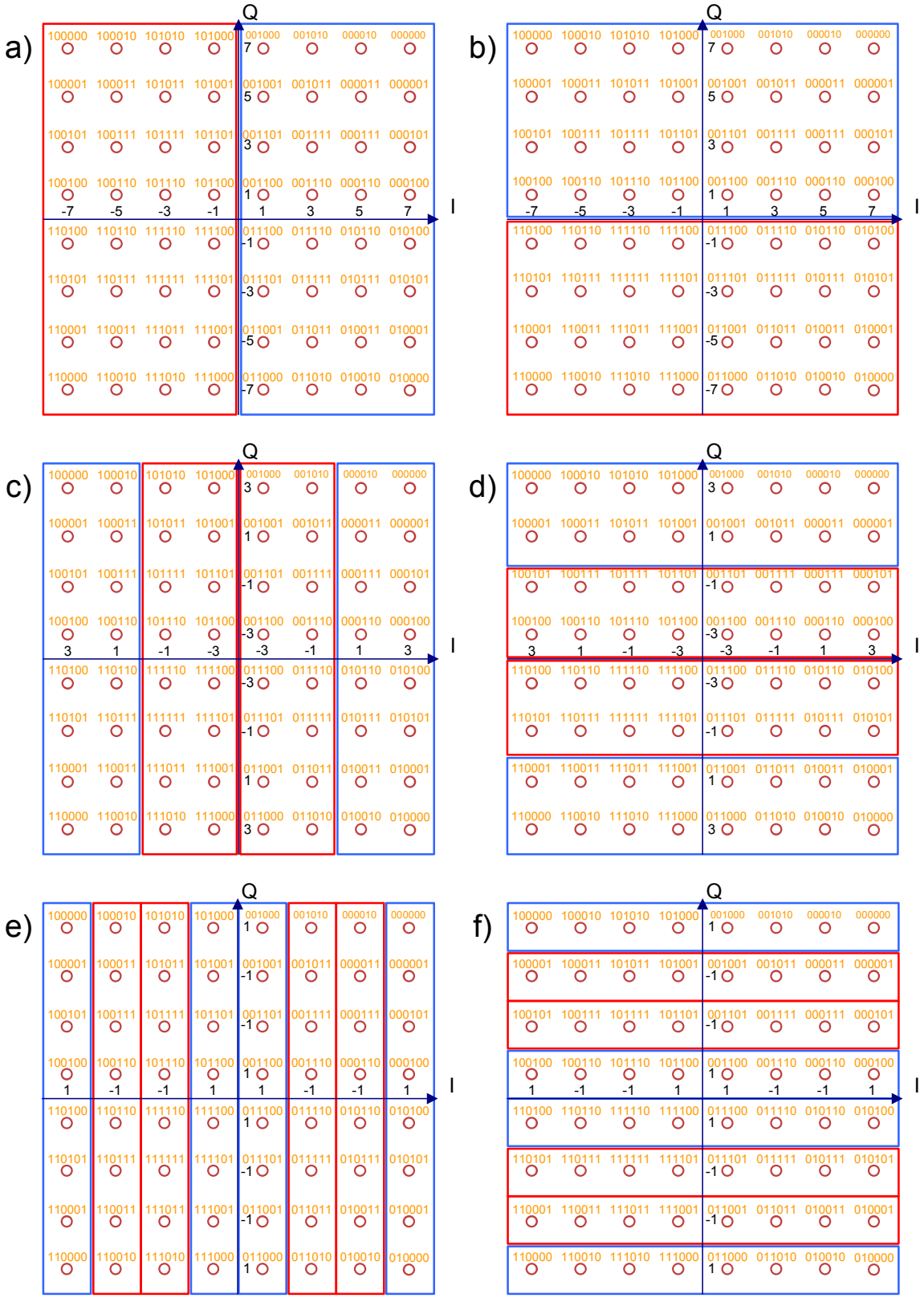

Figura 3.5: Representação gráfica de constelações 64-QAM, sobrepostas pelas áreas utilizadas para a geração de *soft*-bits pelo *demapper*.

Para compatibilizarem-se os valores dos *soft*-bits, como os obtidos no exemplo anterior, com variáveis de um sistema digital mais eficiente, é necessário aplicar-se uma quantização. Quantizar um valor significa alterar proporcionalmente a grandeza que ele representa de forma a poder representá-la em uma variável composta por um limitado número de bits, adequado ao funcionamento do sistema. Para se quantizar um valor é necessário discretizá-lo, dividindo-se a máxima amplitude que ele pode assumir pelo número de intervalos de valores que possam ser expressos pelo número de bits da variável que irá representá-lo. Desta forma um símbolo de uma constelação 64-QAM, como a da Figura 3.5, recebido na posição (3,2 ; 1,4), considerando-se (I_r ; Q_r), seria representado por *soft*-bits expressos por três bits, do seguinte modo:

$$I_{S2} = I_r = 3{,}2$$

Considerando-se que I_{S2} pode variar de -7 até 7 (modulação 64-QAM), então é 14 a máxima variação possível entre um extremo e outro do conjunto representado no eixo dos valores reais. Considerando-se *soft*-bits representados por três bits, este conjunto deverá ser dividido em oito intervalos de decisão. Na Figura 3.6a pode-se ver a representação dos intervalos de decisão para a obtenção dos *soft*-bits, onde um valor "111"$_{\text{BINÁRIO}}$ representa um *soft*-bit com grande certeza de representar o valor "1" booleano, "100"$_{\text{BINÁRIO}}$ representa uma certeza fraca de representar o valor "1" booleano, da mesma forma que "011"$_{\text{BINÁRIO}}$ representa uma maior, porém fraca, certeza de representar o valor "0" booleano e, finalmente "000"$_{\text{BINÁRIO}}$ representando uma forte certeza de representar o valor "0" booleano. Seguindo-se este raciocínio tem-se que o *soft*-bit quantizado de I_{S2} é:

$$I_{SQ2} = "010"_{BINÁRIO}$$

E os demais *soft*-bits deste mesmo símbolo hipotético (3,2 ; 1,4) de uma constelação 64-QAM seriam:

$$Q_{S2} = 1{,}4 \therefore Q_{SQ2} = "010"_{BINÁRIO}$$

$$I_{S1} = |3{,}2| - 4 = -0{,}8 \therefore I_{SQ1} = "100"_{BINÁRIO}$$

$$Q_{S1} = |1{,}4| - 4 = -2{,}6 \therefore Q_{SQ1} = "111"_{BINÁRIO}$$

$$I_{S0} = |-0{,}8| - 2 = -1{,}2 \therefore I_{SQ0} = "111"_{BINÁRIO}$$

$$Q_{S0} = |-2{,}6| - 2 = 0{,}6 \therefore Q_{SQ0} = "001"_{BINÁRIO}$$

Os valores binários acima foram obtidos com base nos intervalos representados nas Figuras 3.6a, para I_{SQ2} e Q_{SQ2}, 3.6b para I_{SQ1} e Q_{SQ1} e 3.6c para I_{SQ0} e Q_{SQ0}.

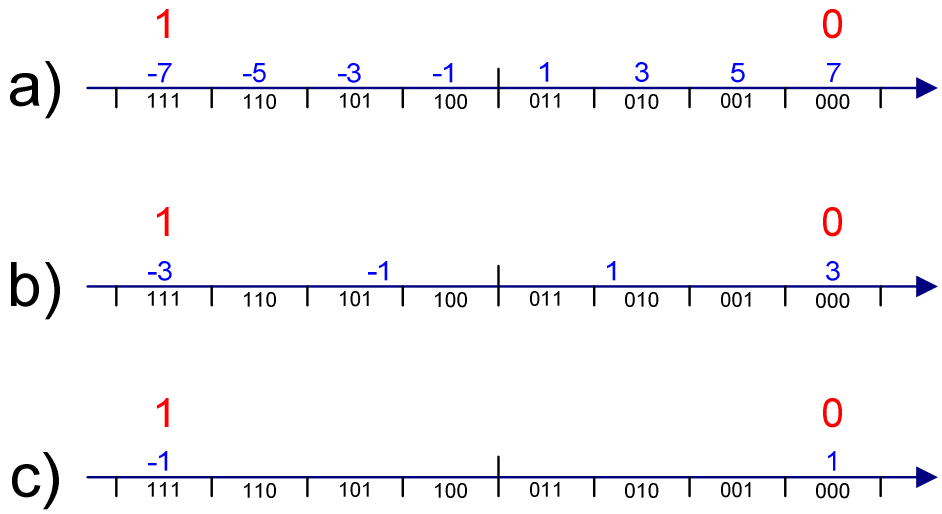

Figura 3.6: Zonas de decisão para obtenção dos *soft*-bits em um dos eixos de uma constelação 64-QAM.

Note-se que para se obter uma decisão do tipo *Hard-Decision* a partir do *demapper* que gera *soft*-bits, basta considerar apenas o bit mais significativo da palavra binária que representa o *soft*-bit.

3.3 Decodificador convolucional

As Figuras 3.7 e 3.8 apresentadas a seguir, representam a lógica e o fluxo de dados do decodificador de Viterbi implementado em software ao qual este trabalho se refere. Nas citadas figuras, há quadrados de cor verde claro dentro de áreas delimitadas por linhas tracejadas também da mesma cor. Nestes quadrados há números, que serão utilizados ao longo de todo este capítulo com o intuito de orientar o leitor quanto a qual das áreas e das funcionalidades do fluxograma o texto estará se referindo.

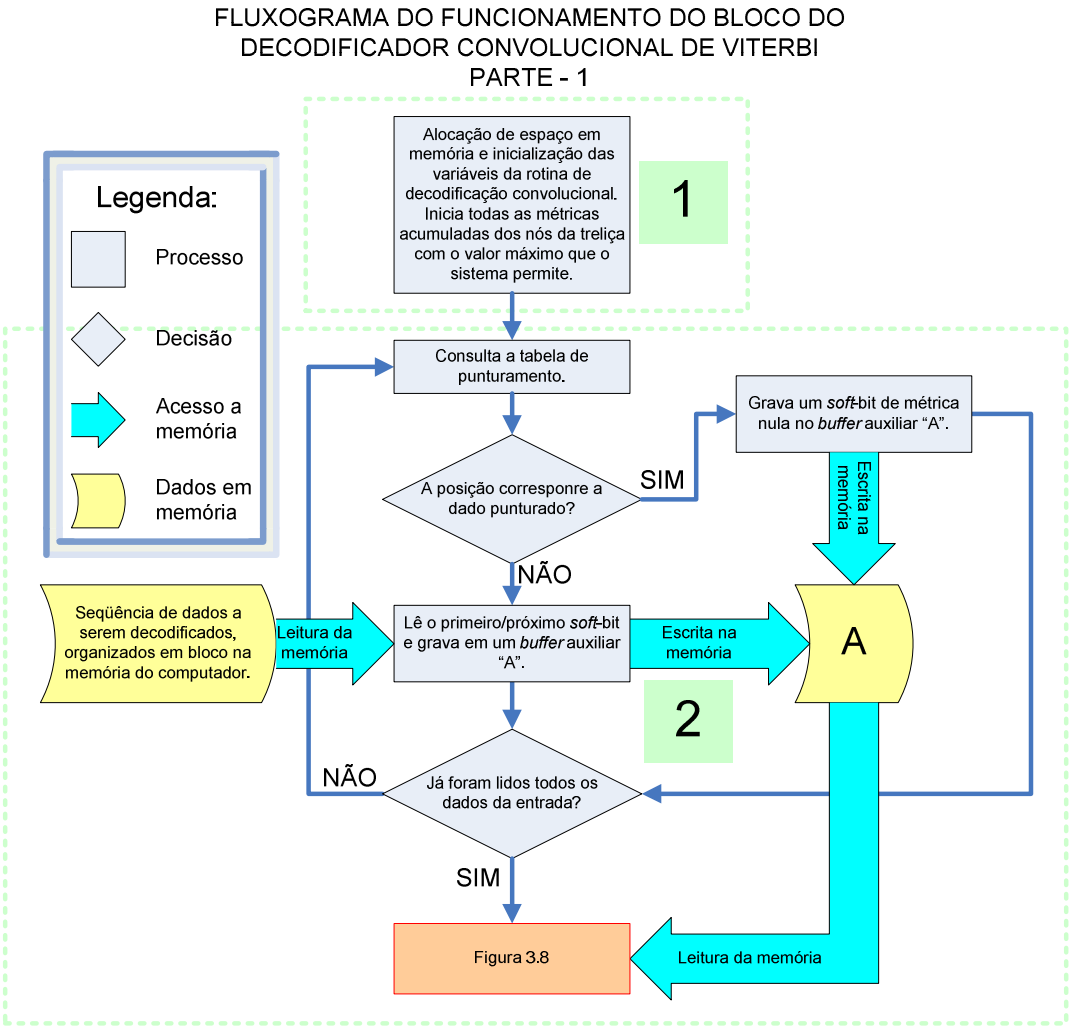

Figura 3.7: Primeira parte do fluxograma representando a lógica do decodificador de Viterbi implementado em software.

Este decodificador convolucional de Viterbi, implementado para simulação em software, da mesma forma que o processo de codificação convolucional apresentado no capítulo 3.1, foi elaborado para tratamento de uma só vez de uma grande massa de dados organizados em blocos de memória. Esta abordagem difere da utilizada na implementação em hardware, em que esta massa de dados "flui" de maneira constante pelo sistema.

O início da execução do algoritmo de simulação do decodificador convolucional de Viterbi está representado na área numerada "1" da Figura 3.7, onde é feita a alocação da memória a ser usada no processo de decodificação e a leitura dos dados de entrada. Também nesta fase as máquinas de estados e os ponteiros usados no processo de decodificação são iniciados.

FLUXOGRAMA DO FUNCIONAMENTO DO BLOCO DO DECODIFICADOR CONVOLUCIONAL DE VITERBI
PARTE - 2

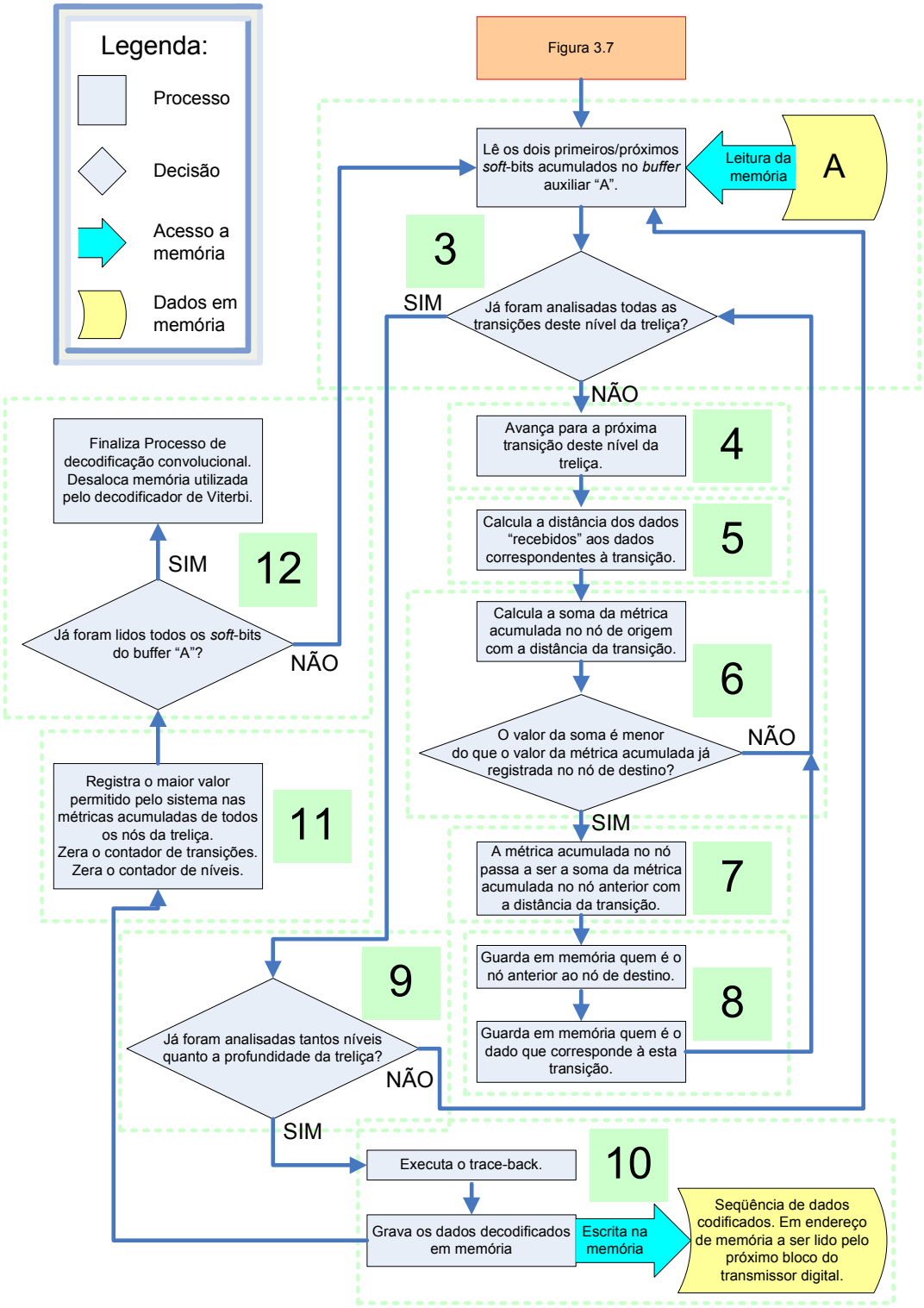

Figura 3.8: Segunda parte do fluxograma representando a lógica do decodificador de Viterbi implementado em software.

A etapa “2”, a seguir, é responsável pelo *depuncturing* dos dados de entrada. Nesta etapa os dados gravados na memória, vindos de processos anteriores da decodificação de canal de um receptor digital, são lidos, passados pelo processo de *depuncturing* e novamente gravados em uma outra área de memória intermediária. O processo de *depuncturing* consiste em acompanhar, via um contador cíclico, a matriz de *puncturing* correspondente ao padrão aplicado aos dados durante a codificação no transmissor digital. Para cada valor 1 constante na matriz de *puncturing*, copia-se um dado contido na seqüência de dados de entrada para a memória intermediária e para cada valor zero, escreve-se um “dummy bit” ou um bit marcado que terá um valor nulo na hora de calcularem-se as distâncias na treliça do decodificador de Viterbi.

Na Figura 3.8, na área “3”, está representado o início da decodificação convolucional em si, em que os dados (bits em formato de *soft*-bits), já passados pelo processo de *depuncturing*, são lidos da memória intermediária e, dois a dois, usados para o processo de decodificação de Viterbi. Cada um destes pares de bits será comparado com cada um dos pares de bits representados por cada uma das possíveis transições em cada um dos níveis da treliça. Nesta etapa, ainda, é feita a verificação se foram calculadas as métricas de todas as possíveis transições do nível da treliça que está sendo processado. Se não, segue o processo para as transições seguintes, se sim, vai para a etapa “9” em que é verificado se todos os níveis foram processados, conforme será visto mais adiante.

A etapa “4” do processo de decodificação, representado na Figura 3.8, está destacada como uma etapa única para ressaltar a forma como foi implementada uma parte do decodificador de *puncturing* Viterbi. Em uma tabela, gravada em memória, foram registradas, em cada linha, cada uma das possíveis transições da máquina de estados do codificador convolucional presente no transmissor digital. Para cada uma das transições há uma coluna com o registro do dado de entrada do codificador, do estado de origem e do estado de destino a que cada transição se refere, bem como os dados codificados resultantes de tal transição. Quando, neste capítulo, fala-se em “ir para a próxima transição”, significa ler-se a próxima linha desta tabela.

Na etapa “5” é feito o cálculo da distância dos dados recebidos pelo receptor digital e os dados resultantes da transição que estiver sendo analisada no momento. Os dados, conforme visto no capítulo anterior, são *soft*-bits que podem variar de valores entre zero e um (com tantos níveis quanto a quantização escolhida permitir). Há inúmeras maneiras de se calcular a distância entre pares de soft-bits, que refletem em decodificações com

diferentes desempenhos na tarefa de corrigir erros. A maneira abordada neste trabalho é a soma simples das distâncias euclideanas entre os bits recebidos e os representados pela transição. Isto é a soma do módulo da diferença entre o cada um dos bits recebidos com o seu respectivo bit representado pela transição que estiver sendo processada.

Na etapa seguinte, “6”, é feita a soma da distância calculada na etapa “5” com o valor da métrica acumulada no nó (estado, ou ponto) de origem da transição que está sendo processada. Ainda nesta etapa é feita a comparação do resultado desta soma com o valor da métrica acumulada no ponto de destino da transição. No caso de a soma resultar em um valor inferior ao previamente registrado como o valor da métrica no ponto de destino, então o valor da métrica acumulada no ponto de destino passa a ser o valor da referida soma. Uma vez que ao se reiniciar uma nova decodificação, ou porque recém iniciou-se o processo no decodificador de Viterbi, ou porque um *trace-back* acabou de ser feito, todos os nós (pontos) da treliça são iniciados com suas métricas acumuladas sendo iguais ao máximo valor permitido pelo sistema. Desta forma, ao final da análise de todas as transições de um nível da treliça, todos os pontos de destino destas transições terão como métricas acumuladas os mesmos valores das métricas dos caminhos de menores métricas acumuladas que neles chegarem, conforme representado na etapa “7” do diagrama da Figura 3.8.

No caso de, na comparação da etapa “6”, detectar-se que a soma da distância entre os bits recebidos e os bits representados pela transição, não ser inferior ao valor da métrica acumulada no ponto de destino, o sistema simplesmente salta novamente para a etapa “3”.

Consultando-se a tabela com as informações sobre as transições, na etapa “8”, obtém-se (quando é registrada uma nova métrica acumulada em um ponto, menor do que a anterior) os ponto de origem e o dado correspondente, na entrada do codificador convolucional, à transição que estiver sento analisada. Estes dados são guardados em memória, em uma posição referenciada ao nó em questão.

Após a execução da etapa “8”, o fluxo do programa é direcionado à etapa “3”, onde, caso não tenham sido analisadas todas as transições de um nível da treliça, é novamente direcionado para a etapa “4” (fechando um ciclo de programa já contemplado nesta explanação), caso contrário o fluxo é redirecionado à etapa “9”.

Na etapa “9” é verificado se foi alcançada a profundidade máxima da treliça, isto é, se foram analisados tantos níveis quanto o determinado para poder-se fazer um *trace-back*. No caso de ainda não terem sido processados todos os níveis da treliça, retorna-se à etapa

"3" onde se inicia o processamento de um novo nível. No caso de a profundidade máxima da treliça ter sido alcançada, é iniciado o processo de *trace-back*.

A etapa "10" do diagrama representa o processo de *trace-back*. Este processo é feito comparando-se e detectando-se, dentre os pontos de destino de todas as transições do último nível da treliça, aquele que apresentar a menor métrica acumulada. A partir deste ponto é feito o rastreamento do caminho de menor métrica acumulada percorrido até chegar-se nele. Para tanto se determina que este ponto seja o ponto presente. A partir deste momento segue-se de maneira recursiva a seguinte rotina:

Resgata-se da memória o dado correspondente à entrada do codificador no transmissor, correspondente à transição do caminho de menor métrica acumulada a chegar ao ponto presente (gravado na memória na etapa "8"). Grava-se este dado em um vetor na memória do computador, como a saída de dados do decodificador convolucional de Viterbi, que será usada por outros processos na cadeia de processos da recepção digital. Resgata-se da memória o ponto anterior à referida transição (gravado na memória na etapa "8"), que passa a ser o ponto presente para a próxima execução desta rotina.

Esta rotina deverá ser executada até que o "caminho de volta" rastreado pelo *trace-back* tenha percorrido todos os níveis da treliça, e, portanto todos os dados ali codificados tenham sido extraídos. Depois de encerrado o *trace-back*, é feita, na etapa "11", uma verificação para saber se todos os dados (*soft*-bits presentes na memória intermediária) de entrada já foram lidos. Em caso afirmativo, os espaços de memória utilizados pelo decodificador de Viterbi são desalocados e a rotina de decodificação convolucional encerrada.

No caso de ainda haverem dados na entrada do sistema para serem decodificados, o fluxo de execução do programa de decodificação de Viterbi é desviado para a etapa "12" onde a treliça é reiniciada, zerando-se os contadores de níveis de treliça processados e de transições processadas. Ainda nesta etapa são novamente gravados os valores máximos permitidos pelo sistema nas métricas acumuladas de cada um dos nós da treliça. Após reiniciada a treliça o programa é redirecionado para a etapa "3", onde se dá continuidade a todo o processo já descrito anteriormente neste capítulo.

3.3.1 *Soft-Decision*

A principal característica que diferencia o método de decodificação por *Soft-Decision*, é a utilização das distâncias euclidianas entre os símbolos recebidos e as

posições de referência dos símbolos da constelação, para o cálculo das métricas nos pontos da treliça. Uma vez que, como visto anteriormente, os dados de entrada do decodificador de Viterbi com *Soft-Decision*, para sistemas M-QAM com intercalação de bits, devem ser do tipo *soft*-bits, será descrita a forma como estes dados são utilizados para a determinação destas métricas.

Para cada nível da treliça analisado, há dois *soft*-bits que fazem parte da seqüência de dados recebidos e dois bits, por transição de estados, com os quais eles serão comparados. Os bits referentes às transições são convertidos para *soft*-bits de valores que representem absolutamente um ou absolutamente zero ("111" e "000", respectivamente para o caso de utilizarem-se três bits para representar um *soft*-bit) para compatibilizá-los com os *soft*-bits da entrada de dados. A comparação da distância, assim como na distância de *Hamming* (no caso do *Hard-Decision*), é, via de regra, feita elemento com elemento correspondente, dos dois pares de bits. Quatro diferentes métodos de cálculo de métricas nos pontos foram testados onde b_{r0} e b_{r1} são os dois *soft*-bits recebidos, b_{c0} e b_{c1} são os *soft*-bits de cada uma das transições e M a métrica do caminho no ponto:

$$M = b_{r0} \times (-b_{co}) + b_{r1} \times (-b_{c1}) \tag{1}$$

$$M = |b_{r0} - b_{c0}| + |b_{r1} - b_{c1}| \tag{2}$$

$$M = \sqrt{|b_{r0} - b_{c0}|^2 + |b_{r1} - b_{c1}|^2} \tag{3}$$

$$M = |b_{r0} - b_{c0}|^2 + |b_{r1} - b_{c1}|^2 \tag{4}$$

Destes métodos, o que apresentou melhores resultados foi o (4), sendo este o usado nas comparações que serão apresentadas no capítulo de resultados deste trabalho.

3.3.2 *Hard-Decision*

Para o cálculo da métrica acumulada no ponto, quando utilizando-se do método *Hard-Decision* basta somar-se o valor da diferença, um a um dos elementos correspondentes dos dois pares de bits, dos bits recebidos e dos bits correspondentes à transição que estiver sendo analisada. Para fácil aproveitamento do código já escrito para a decodificação *Soft-Decision*, bem como do *demapper*, adaptou-se o código para que este se comportasse exatamente da forma que um decodificador de Viterbi escrito puramente para o método *Hard-Decision* se comportaria.

Para adaptar-se o decodificador de Viterbi por *Soft-Decision* para funcionar como um decodificador por *Hard-Decision*, bastou analisar o sinal dos *soft*-bits. Quando o sinal de um *soft*-bit recebido diferencia-se do *soft*-bit da transição analisada, soma-se um à distancia de *Hamming* acumulada no ponto, caso contrário não soma-se nada. A partir deste ponto, a utilização das distâncias de *Hamming* acumuladas nos caminhos passa a ser idêntica às métricas obtidas pelo método *Soft-Decision*, eliminando-se os caminhos de maiores métricas acumuladas que chegam aos pontos e, por fim, fazendo-se o *trace-back* sobre o caminho sobrevivente de menor métrica acumulada.

3.4 A integração

Os códigos ANSI C, gerados para a simulação do codificador convolucional e do decodificador convolucional de Viterbi, cujos desenvolvimentos são descritos neste trabalho, foram integrados a simuladores dos sistemas DVB-T e ISDB-T completos. Porém, antes desta integração, foram feitos testes de validação dos quais os resultados serão apresentados no capítulo 4.

Para a etapa de validação dos simuladores, do codificador e do decodificador convolucionais, foi elaborado um sistema de transmissão digital simplificado em que apenas as etapas de codificação e decodificação convolucionais estão contidas na codificação de canal (Figura 3.9). Os dados que entram no sistema são dados binários conhecidos. Os dados da saída do codificador convolucional com *puncturing* são mapeados para 16 ou 64-QAM e representados por um par de variáveis do tipo *float* (ponto flutuante) que formam o par complexo I e Q. Estes pares de I e Q são passados para um programa externo que simula a ação degradante de um canal aditivo gaussiano e os devolve no mesmo formato, porém com valores distorcidos pela ação do canal. Por fim, os dados "vindos do canal" são a entrada de um simulador de *demapper* com dados de saída do tipo *soft*-bits, que são, por sua vez, a entrada do bloco dos *depuncturing* e decodificador convolucional de Viterbi.

Para finalizar a integração do sistema de comunicação simplificado que visa a validação dos blocos de codificação e decodificação convulsionais, foi elaborado um programa comparador. Este programa é responsável por comparar e contabilizar as diferenças (erros) entre o arquivo de dados que entrou no codificador convolucional com

os dados da saída do decodificador convolucional de Viterbi, gerando, como saída, um relatório.

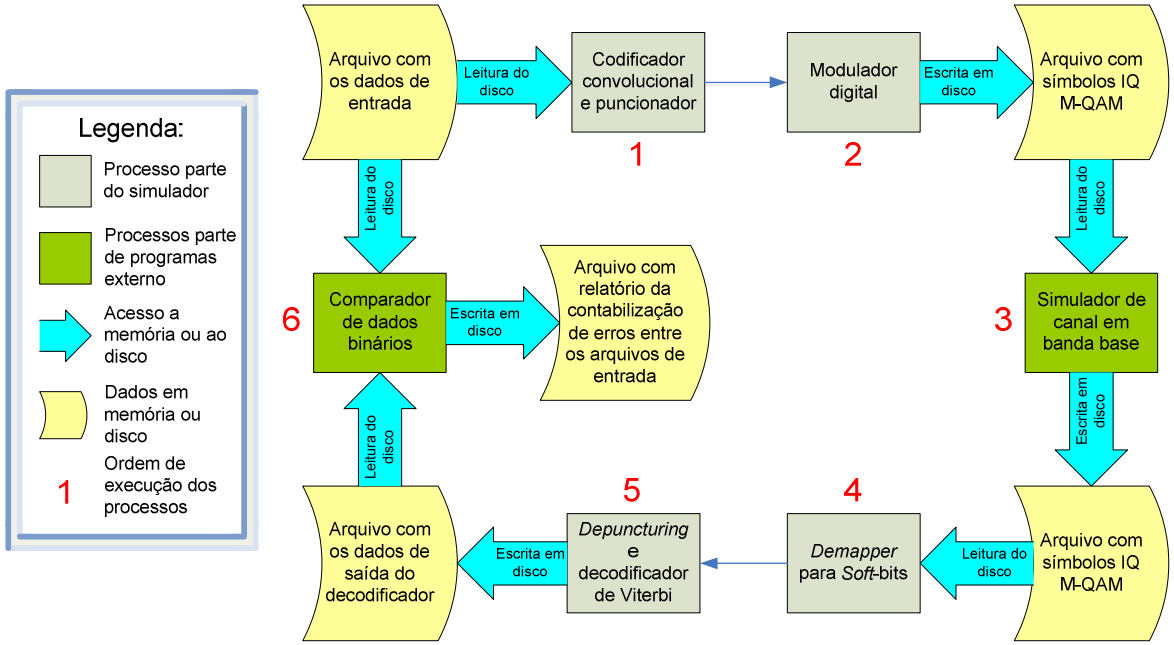

Figura 3.9: Sistema simplificado de comunicação digital para teste e validação dos simuladores de codificação e decodificação convolucionais.

4 Resultados

O presente trabalho tem como objetivo a validação dos métodos de implementação prospectados para a construção de um mecanismo de codificação e decodificação convolucionais compatível com o funcionamento de um sistema de comunicação de dados digital que se utilize de modulação M-QAM e intercalação de bits. A referida validação foi feita utilizando-se do sistema simplificado descrito no capítulo 3.

Para a geração dos dados utilizados para a análise e validação dos métodos utilizados na concepção dos simuladores a que este trabalho se refere, foi elaborado um *script* capaz de configurar e rodar automaticamente e na ordem correta o conjunto de softwares de simulação pelo número de vezes desejado. A cada iteração deste *script*, as configurações de SNR do simulador de canal aditivo gaussiano são alteradas, de forma a gerar dados de saída representando a variação no desempenho do CODEC convolucional dentro de uma faixa de situações equivalentes às possivelmente encontradas em situações reais de funcionamento de um sistema de comunicação para *broadcast* digital.

Foram rodados conjuntos de simulações para modulações 16-QAM e para 64-QAM. Para cada um dos tipos de modulação foram rodados subconjuntos de testes para os modos de funcionamento do decodificador convolucional de Viterbi, *Hard-Decision* e *Soft-Decision*. Os resultados apresentados neste capítulo para o funcionamento do decodificador em modo *Soft-Decision*, dizem respeito apenas ao método de cálculo das distâncias nos pontos que apresentou melhor desempenho, conforme já citado no capítulo 3.

Mais precisamente foram rodadas 40 iterações para cada modo de operação do decodificador de Viterbi, para cada uma das duas modulações QAM. Em cada iteração o mesmo conjunto de dados foi passado pelo sistema simulador, com o simulador de canal tendo sua configuração de SNR incrementada de um a partir do valor zero a cada iteração. Portanto, para cada uma das possíveis configurações do CODEC convolucional, foi feito um levantamento do seu desempenho, quanto à capacidade de correção de erros, sobre canais aditivos gaussianos com SNR na faixa de 0dB a 39dB (os gráficos de desempenho a

seguir, apenas representam o intervalo de SNR onde o número de erros for superior à zero). Todos os resultados de simulações foram obtidos utilizando-se de taxas de codificação convolucional de ½, a mesma descrita na literatura [1][2][3] de onde obteve-se os valores de desempenho teóricos de referência que procurou-se atingir com o decodificador de Viterbi desenvolvido neste trabalho.

Após cada iteração dos simuladores, codificador e decodificador convolucionais e de canal, um programa comparador é rodado. O programa comparador conta o número de bits diferentes entre o arquivo de entrada do sistema e o arquivo gerado com os dados da saída do decodificador de Viterbi. Depois da contabilização das diferenças feita, é obtida a razão do número de bits errados pelo número de bits "enviados". Esta razão é chamada de taxa de erro de bits, ou *Bit Error Rate*, ou ainda BER. A BER de cada iteração é então gravada em um arquivo, que é gerado para cada diferente configuração do CODEC convolucional. A partir dos arquivos de BER foram traçados gráficos de desempenho, possibilitando a comparação visual das características de cada um dos diferentes modos de operação do decodificador de Viterbi.

A seguir, na Figura 4.1, é apresentado um gráfico do desempenho do decodificador convolucional de Viterbi para modulação 16-QAM. No eixo das abscissas está representada a SNR aplicada pelo simulador de canal para cada uma das iterações do sistema. No eixo das ordenadas estão representados os valores de BER em escala logarímica. Neste gráfico estão representados dois conjuntos de dados, o "H16", representando os dados obtidos com o decodificador de Viterbi operando no modo *Hard-Decision* e o "K16 1-1" representando os dados obtidos com o decodificador de Viterbi operando no modo *Soft-Decision*.

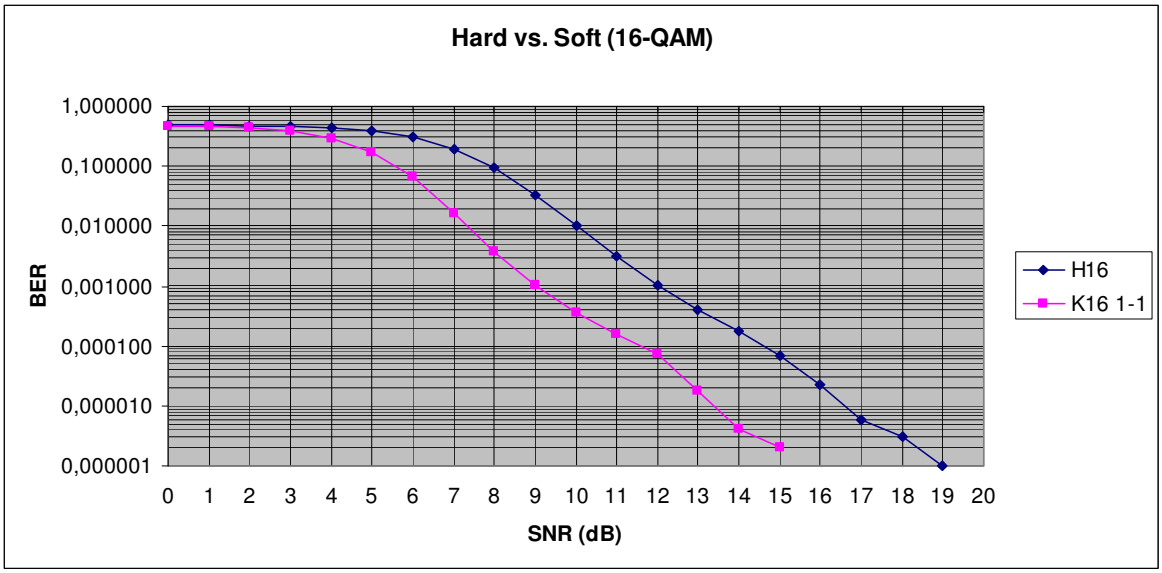

Figura 4.1: Gráfico de desempenho do decodificador convolucional de Viterbi operando nos modos *Hard* e *Soft-Decision* para modulação 16-QAM.

Na Figura 4.2, é apresentado um gráfico do desempenho do decodificador convolucional de Viterbi para modulação 64-QAM nos mesmos moldes do apresentado na Figura 4.1. Neste gráfico estão representados dois conjuntos de dados, o "H64", representando os dados obtidos com o decodificador de Viterbi operando no modo *Hard-Decision* e o "K64 1-1-1" representando os dados obtidos com o decodificador de Viterbi operando no modo *Soft-Decision*.

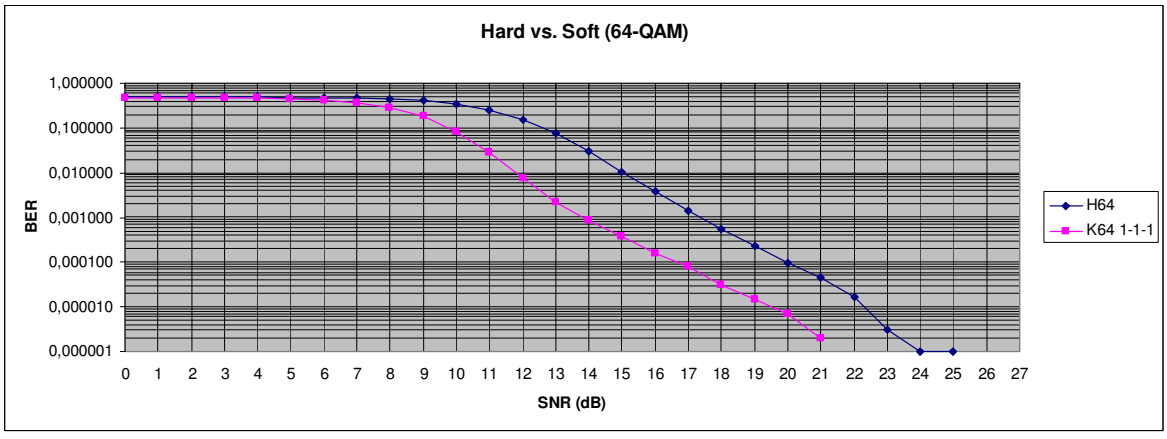

Figura 4.2: Gráfico de desempenho do decodificador convolucional de Viterbi operando nos modos *Hard* e *Soft-Decision* para modulação 64-QAM.

É possível observar nas Figuras 4.1 e 4.2 que o desempenho de ambos os decodificadores convolucionais, tanto do operando em modo *Hard-Decision* quanto do em *Soft-Decision*, é inferior quando decodificando dados transmitidos através do canal

utilizando-se de uma modulação 64-QAM em relação à 16-QAM. Isto deve-se ao fato de constelações mais densas terem seus símbolos efetivamente mais próximos uns dos outros, facilitando que haja a "invasão" dos símbolos sobre a área de decisão de símbolos adjacentes. Estas "invasões" acabam por sobrecarregar o decodificador em níveis de SNR inferiores em relação a constelações menos densas, e têm como conseqüência a diminuição da capacidade de correção de erros.

Por outro lado, em ambas as figuras é possível observar uma diferença entre o desempenho dos decodificadores convolucionais que operam em modos *Hard* e *Soft-Decision*, que são equivalentes quando comparadas entre as diferentes modulações M-QAM. Tal diferença demonstra justamente a diferença de desempenho entre os modos de operação do decodificador de Viterbi. Praticamente não há diferença de desempenho em níveis de SNR em que os decodificadores, operando em ambos os modos, não têm a capacidade de corrigir erros, devido ao altíssimo nível de ruído adicionado sobre o nível do sinal transmitido.

Porém, uma vez alcançados níveis de SNR em que os decodificadores, operando em ambos os modos, passam a funcionar com sua capacidade plena de correção de erros (as curvas de desempenho tornam-se descendentes com inclinações aproximadamente constantes), pode-se observar níveis semelhantes das capacidades de correção de erros (BER) para níveis de SNR distintos de até 3dB entre as curvas. Pode-se então dizer que a capacidade de correção de erros do decodificador de Viterbi, desenvolvido para o simulador deste trabalho, tem um desempenho 3dB maior, quando operando em modo *Soft-Decision* do que quando operando em modo *Hard-Decision*.

5 Conclusão

A codificação convolucional é uma maneira de correlacionar dados enviados através do canal com aqueles dados enviados imediatamente antes e após, através de uma máquina de estados finitos com uma estrutura de memória do tipo *shift-register*. Os dados são passados através deste *shift-register*, enquanto diferentes operações lógicas entre os elementos desta memória geram uma saída de dados dependentes do seu conteúdo. O fato de o codificador ser uma máquina de estados finitos, faz com que mudanças de estados estejam sempre vinculadas ao conteúdo da memória e ao próximo dado da seqüência a ser transmitida.

O ruído aditivo gaussiano, que tem a característica de se somar aos sinais que se propagam no meio do canal, adiciona valores de amplitude que não apresentam correlação entre si ao longo do tempo. “Conhecendo” a máquina de estados, presente no transmissor, que gerou a seqüência de dados recebidos e que apenas permite transições específicas entre diferentes estados, o decodificador de Viterbi é capaz de detectar a seqüência de dados originalmente codificados, através de um processo de escolha de possíveis seqüências por meio de análise de máxima verossimilhança. Sendo possível assim recuperar a seqüência de dados codificados e corrigir erros eventualmente inseridos na transmissão dos dados através do canal.

O método *Soft-Decision* de decodificação de Viterbi procura quantificar o erro entre cada símbolo recebido com a posição de referência dos símbolos das constelações M-QAM (do mesmo modo para outros tipos de constelação). Desta forma é possível, para o *demapper,* gerar mais informação, que permita ao decodificador de Viterbi fazer uma “escolha” mais apurada dentre as possíveis seqüências de dados que teriam sido geradas no transmissor e que mais se aproximem da seqüência recebida no receptor. Enquanto o método *Hard-Decision* apenas utiliza informações que indiquem se um dado binário recebido foi interpretado pelo *demmaper* como sendo de valor um ou de valor zero, no método *Soft-Decision* os dados de entrada do Viterbi trazem informações com um grau de

certeza, podendo um símbolo estar transportando um bit com grande ou pequeno grau de certeza de ter sido transmitido com valor um ou zero.

Há vários métodos de se extrair dos símbolos os erros que a eles foram inseridos pelo ruído aditivo gaussiano do canal, atribuindo-se aos bits que eles representam valores em um intervalo subdividido entre zero e um. Porém, a "extração" deste erro e a sua representação nos conjuntos de bits a serem decodificados pelo decodificador de Viterbi, deixam de ser uma operação óbvia quando os bits "transportados" em um símbolo são separados antes da decodificação convolucional pelo *deinterleaver* de bits. O bit *interleaving* é uma maneira de aumentar o desempenho do decodificador de Viterbi, prevenindo contra erros causados por *bursts* interferentes do canal. Ela evita a concentração de bits errados na seqüência de dados recebidos, concentração esta capaz de fazer com que seja excedida a capacidade de correção do decodificador convolucional, podendo fazê-lo gerar mais erros do que corrigi-los.

A principal conseqüência do processo de bit *interleaving* e bit *deinterleaving* é o fato de os bits de dados seqüenciais serem transmitidos através de símbolos diferentes. Isto gera uma situação singular no instante de atribuir uma representação do erro do posicionamento do símbolo, em relação à constelação de referência, aos bits a serem decodificados. A forma utilizada para resolver esta situação é a utilização dos *soft*-bits, que permite atribuir à cada bit a parcela que lhe cabe do erro de posicionamento do símbolo que o "transportou". A geração deste tipo de dados e a sua correta utilização para a decodificação em um decodificador de Viterbi com *Soft-Decision* foi o objetivo dos estudos que originaram este trabalho, e a motivação foi a demanda interna dos projetos de P&D em execução no CPTW aliada à dificuldade de se encontrar materiais didáticos e científicos que orientassem o desenvolvimento de um sistema com as características de desempenho desejadas.

A partir das dificuldades em obter as informações necessárias à implementação do sistema de codificação e decodificação convolucionais que atendesse à demanda do CPTW, desenvolveu-se em linguagem ANSI C um simulador para um codificador convolucional e um decodificador de Viterbi com as características procuradas, de forma a validar o compilado de dados obtidos através de pesquisa bibliográfica e de inferências feitas através dela. Este trabalho foi feito durante bolsa de estágio vinculada ao curso de mestrado em engenharia elétrica com ênfase em tecnologia da informação. Havia a necessidade do desenvolvimento de um decodificador de Viterbi e as informações

necessárias para guiar a execução desta tarefa eram incompletas, nem sempre úteis e, por muitas vezes, inexistentes. Desta forma, a linha que se procurou seguir durante a escrita desta dissertação foi a de buscar trazer informações que pudessem ser da melhor forma possível, utilizadas e reutilizadas em trabalhos a serem desenvolvidos futuramente no CPTW.

Os resultados obtidos, descritos e graficamente apresentados no Capítulo 4, de uma diferença próxima de 3dB no desempenho dos dois decodificadores de Viterbi, um funcionando em modo *Hard-Decision* e outro em modo *Soft-Decision*, são compatíveis com aqueles citados no Capítulo 2. Esta diferença de 3dB no desempenho é o valor esperado teórico para decodificadores operando nestes dois modos distintos. Uma vez que estes dados apresentados no Capítulo 2 são baseados em informações contidas em referências [1][2][3] de conhecido renome, em se tratando de publicações especializadas no assunto de codificação de sinais para sistemas de comunicação digital, considerou-se os resultados obtidos validados.

Outro resultado que converge para a idéia de que o trabalho de pesquisa e desenvolvimento do modo de implementação de um decodificador de Viterbi com S*oft-Decision* e bit *interleaver* atingiu os objetivos a que se propôs, foi a simplicidade computacional da versão final do algoritmo de decodificação implementado. Esta simplicidade se reflete na pequena diferença de complexidade entres as versões *Soft* e *Hard* do modo de operação deste decodificador, que corresponde aos argumentos, apresentados na introdução deste trabalho, de que o aumento na complexidade computacional é inteiramente compensado pelo aumento na capacidade de correção de erros do modo *Soft-Decision*.

Como continuidade da pesquisa descrita neste trabalho, será feita a implementação do algoritmo aqui descrito em hardware de lógica programável. Desta forma será possível a integração efetiva do codec convolucional de Viterbi com *soft-decision* aos projetos desenvolvidos no CPTW que têm como objetivo final a criação de chips dedicados, voltados a sistemas de comunicação digitais. Para tanto, os resultados obtidos nesta pesquisa serão passados aos bolsistas de iniciação científica do CPTW para que estes possam usá-los como ponto de partida para a descrição do *hardware*, assim como parâmetro de comparação para os testes de desempenho dos códigos a serem gerados.

6 Referências Bibliográficas

[1] R. H. Morelos-Zaragoza, *The Art of Error Correcting Coding*, John Wiley & Sons Ltd, julho 2004.

[2] J. G. Proakis, *Digital Comunications*, 4ª edição, McGraw-Hill, 2001.

[3] B. Sklar, *Digital Communications Fundamentals and Applications*, 2ª edição, Prentice Hall, 2001.

[4] P. Sweeney, *Error Control Coding From Theory to Practice*, John Wiley & Sons Ltd, março 2004.

[5] F. Tosato e P. Bisaglia, "*Simplified Soft-Output Demapper for Binary Interleaved COFDM with Application to HIPRLAN/2*", Communications, 2002. ICC 2002. IEEE International Conference on Publication, 2002, Volume: 2, On page(s): 664-668 vol.2.

[6] G. Caire, G.Taricco e E. Biglieri, "*Bit-Interleaved Coded Modulation*", Information Theory, IEEE Transactions on Publication, May 1998, Volume: 44, Issue: 3, On page(s): 927-946.

[7] A. J. Viterbi, J. K. Wolf, E.Zehavi e R. Padovani, "*A Pragmatic Approach to Trellis-Coded Modulation*", Communications Magazine, IEEE Publication, Jul 1989, Volume: 27, Issue: 7, On page(s): 11-19.

[8] E.Zehavi, "*8-PSK Trellis Codes for a Rayleigh Channel*", Military Communications Conference, 1989. MILCOM '89. Conference Record. 'Bridging the Gap. Interoperability, Survivability, Security'., 1989 IEEE Publication, 15-18 Oct 1989, On page(s): 536-540, vol.2.

[9] Y. Wang, JH. Ge Bo Ai, P. Liu e SY. Yang, "*A Soft Decision Scheme for Wireless COFDM with Application to DVB-T*", Consumer Electronics, IEEE Transactions

on Publication, Feb 2004, Volume: 50, Issue: 1, On page(s): 84- 88.

[10] X. Li e J.A.Ritcey, "*Bit-Interleaved Coded Modulation with Iterative Decoding*", Communications, IEEE Transactions on Publication, Aug 2002, Volume: 50, Issue: 8, On page(s): 1250- 1257.

[11] T.C.Hewavithana e M. Brookes, "*Soft Decision for DQPSK Demodulation for the Viterbi Decoding of Convolutional Codes*", Communications, IEEE Transactions on Publication, Jun 1999, Volume: 47, Issue: 6 On page(s): 837-843.

[12] K.Barman, K.Sriram e V.U.Reddy, "*Performance of a Sub-Optimal Metric for Soft-Decision Viterbi with Bit Interleaving in OFDM Based Systems*", Communications, IEEE Transactions on, Feb 1998, Volume: 46, Issue: 2, On page(s): 182-190.

[13] E.Lüthi, Emmanuel Casseau, "*High Soft Output Viterbi Decoder*", European Design and Test Conference, 1996. ED&TC 96. Proceedings, 11-14 Mar 1996, On page(s): 315-319.

[14] Dr.G.Edwards, "*A 45-Mbits/séc. VLSI Viterbi Decoder for Digital Video Applications*", Telesystems Conference, 1993. 'Commercial Applications and Dual-Use Technology', Conference Proceedings., National, 16-17 Jun 1993, On page(s): 127-130.

[15] G.Ungerboeck, "*Channel Coding with Multilevel/Phase Signals*", Information Theory, IEEE Transactions on, Jan 1982, Volume: 28, Issue: 1, On page(s): 55-67.

[16] M.Hosemann, R.Habendorf e G.P.Fettweis, "*Hardware-Software Codesign of a 14.4Mbit–64 State–Viterbi Decoder for an Application-Specific Digital Signal Processor*", Signal Processing Systems, 2003. SIPS 2003. IEEE Workshop on Publication, 27-29 Aug. 2003, On page(s): 45- 50.

[17] J.B.Cain, G.C.Clark e J.M.Geist, "*Punctured Convolutional Codes of Rate (n-1)/n and Simplified Maximum Likelihood Decoding*", Communications, IEEE Transactions on, Apr 1988, Volume: 36, Issue: 4, On page(s): 389-400.

[18] ETSI, "*Digital Video Broadcasting (DVB): Framing, channel coding and modulation for digital terrestrial television*", ETSI EN300 744 V1.5.1, Novembro

2004.

[19] ARIB, "*Transmission System for Digital Terrestrial Television Broadcasting*", STD-B31 Version 1.5, Julho 2003.