1 - Introdução à Compressão Multimédia

A codificação e representação de informação multimédia é uma área tecnológica em expansão. As aplicações multimédia combinam conteúdos que pertencem a tipos de informação digital de natureza heterogénea (também designados por tipos de média), incluindo o texto, os gráficos vectoriais, as imagens *bitmap*, o vídeo digital, o áudio digital e a animação (Ribeiro, 2004). Nos últimos anos, os formatos de codificação e representação destes tipos de informação multimédia têm evoluído em termos das técnicas a que recorrem para permitir a respectiva utilização em aplicações multimédia. Muitos destes formatos deram igualmente origem a normas internacionais.

Quando se considera a utilização destes tipos de informação multimédia é necessário ponderar os recursos de armazenamento e/ou de largura de banda que serão consumidos, uma vez que ambos se tratam de recursos finitos e escassos. Assim, a utilização de tipos de média como o áudio e o vídeo requer, por um lado, que a informação seja comprimida e, por outro, que se preserve tanto quanto possível a qualidade da informação original, isto é, fazendo com que a informação que se perde durante o processo de compressão seja aquela que é a mais irrelevante do ponto de vista do utilizador humano.

Neste contexto, este livro tem como objectivo principal fornecer uma visão aprofundada, fundamentada, objectiva e actualizada das tecnologias de compressão multimédia, permitindo compreender as técnicas que estão por trás dos formatos de representação dos média e que possibilitam a utilização de conteúdos multimédia nos vários tipos de aplicações multimédia actuais, em particular dos tipos de informação multimédia que normalmente se classificam como tipos de média não-estruturados: áudio, imagem e vídeo. Este livro, surgindo na sequência da obra *Multimédia e Tecnologias Interactivas* (Ribeiro, 2004), pretende aprofundar a compreensão dos métodos de compressão, com perdas e sem perdas, que foram introduzidos nesse livro. Assim, analisam-se com detalhe os princípios de funcionamento e os algoritmos de compressão utilizados, entre outros, nos formatos JPEG (*Joint Photographic Experts Group*) e JPEG2000 para a codificação de imagens *bitmap*, e nos formatos da família MPEG (*Moving Picture Experts Group*), nomeadamente nas normas MPEG-1, MPEG-2, MPEG-4 e MPEG-7, abrangendo necessariamente as partes das normas que contemplam a codificação quer de áudio digital, incluindo o MP3, o AAC (*Advanced Audio Coding*) e outras técnicas tais como o ADPCM (*Adaptive Differential Pulse-Code Modulation*) e o AC-3, quer de vídeo digital.

Neste capítulo, faz-se uma introdução às técnicas de compressão mais relevantes no contexto da codificação de informação multimédia. O objectivo é identificar e analisar os mecanismos principais que são utilizados nos algoritmos de compressão em que se baseiam os formatos e normas de codificação de áudio, imagem e vídeo.

© FCA - Editora de Informática

1.1 MOTIVAÇÃO

A compressão, em particular de áudio, imagem e vídeo digitais, é necessária por dois motivos principais:

- Para limitar o espaço de armazenamento consumido pelos conteúdos multimédia;
- Para possibilitar a transferência de conteúdos multimédia sobre redes de comunicações utilizando as taxas de transferência, também designadas por débitos binários ou *bit rates*, existentes actualmente.

Para compreender melhor a necessidade de métodos de compressão de informação multimédia examinemos os seguintes exemplos.

Consideremos em primeiro lugar a quantidade de dados contidos numa imagem *bitmap* correspondente a uma *frame* de vídeo PAL (*Phase Alternating Line*) integral (Ribeiro, 2004). Por exemplo, para armazenar uma imagem com uma resolução de 768×576 píxeis (VGA) e uma profundidade de cor de 24 bits por píxel (codificação RGB com 8 bits por componente) necessitamos de aproximadamente 1296 Kbytes, isto é 1,265 Mbytes, já que:

$$\text{Número de píxeis} = 768 \times 576 = 442368 \text{ píxeis}$$

$$442368 \text{ píxeis} \times 24 \text{ bpp} = 10616832 \text{ bits} = \frac{\frac{10616832}{8} bytes}{1024} = 1296 \text{ Kbytes}$$

Se considerarmos que cada uma destas imagens é efectivamente uma *frame* de uma sequência de vídeo digital PAL, é possível determinar a **largura de banda** necessária para transmitir um fluxo contínuo destas tramas de vídeo a 25 FPS numa rede como sendo de 253,125 Mbit/s, já que:

$$1296 \text{ Kbytes} \times 25 \text{ FPS} = 32400 \text{ Kbyte/s} = \frac{32400 \times 8 \text{ bits}}{1024} = 253{,}125 \text{ Mbit/s}$$

Se considerarmos os requisitos de **armazenamento** impostos por esta sequência de vídeo digital PAL integral, por exemplo num DVD-ROM (capacidade de 4,377 Gbytes), é possível determinar que um disco óptico permitiria apenas armazenar 141,65 segundos desta sequência de vídeo (aproximadamente 2 minutos e meio), já que:

$$\frac{4{,}377 \times 1024 \times 1024 \text{ Kbytes}}{32400 \text{ Kbyte/s}} = 141{,}65 \text{ s} = 2{,}36 \text{ min}$$

© FCA - Editora de Informática

Assim, recorrendo a um disco rígido magnético, e supondo que esta sequência de vídeo digital corresponde a um filme com 90 minutos, é possível determinar que necessitaríamos de 166 Gbytes de espaço de armazenamento em disco para armazenar este filme, dado que:

$$32400\,\text{Kbytes} \times 90\,\text{min} \times 60\,\text{s} = 174960000\,\text{Kbytes} = 166{,}85\,\text{Gbytes}$$

Estes cálculos podem facilmente ser repetidos para determinar o espaço de armazenamento e a largura de banda consumidos por outros tipos de média. Por exemplo, se considerarmos a digitalização de uma fotografia de 35 mm a cores por intermédio de um *scanner* com uma resolução média de 2000×2000, isto corresponde a reter apenas 4 milhões dos 20 milhões de píxeis que constituem a fotografia analógica original. Esta digitalização resulta na geração de um ficheiro com aproximadamente 10 Mbytes. Como é evidente, caso se aumente a resolução da digitalização, qualquer fotografia de 35 mm poderá facilmente ocupar 50 Mbytes de espaço de armazenamento. Isto verifica-se, por exemplo, em aplicações médicas usadas num hospital onde se verifica a necessidade de armazenar e disponibilizar para consulta dezenas de milhões de digitalizações de raios X (Fluckiger, 1995).

Do que ficou exposto, torna-se evidente a necessidade de reduzir a quantidade de dados envolvida na reprodução de áudio digital, imagens *bitmap* ou vídeo digital. Neste contexto, a compressão de informação permite:

- Reduzir o espaço de armazenamento exigido pelos conteúdos multimédia;
- Aumentar a velocidade do acesso aos conteúdos multimédia.

É, pois, possível concluir que a compressão apresenta-se como a única forma existente para se conseguir armazenar, disponibilizar e transmitir as grandes quantidades de dados exigidas pelos conteúdos multimédia, por exemplo, para permitir manipular e visualizar vídeo digital num PC.

A compressão de dados designa um processo pelo qual se converte um fluxo de dados de entrada (fluxo de dados original) num outro fluxo de dados que contém dados comprimidos, ocupando menos espaço de armazenamento. Neste contexto, um fluxo de dados tanto pode ser um ficheiro como um *buffer* de memória. Quando se descomprime o fluxo de dados comprimido, obtém-se um fluxo de dados de saída, que estão descomprimidos e que podem, ou não, ser idênticos aos dados contidos no fluxo de dados original, conforme se trate de uma técnica de compressão sem perdas ou com perdas, como se verá mais adiante. A Figura 1.1 ilustra o diagrama de blocos de um esquema de compressão genérico. Os esquemas de compressão podem igualmente ser designados por técnicas de compressão. Como se pode verificar, a compressão é realizada por um codificador e a descompressão por um descodificador; à saída do codificador encontramos os códigos (também designados por *codewords*) correspondentes à informação comprimida.

© FCA - Editora de Informática

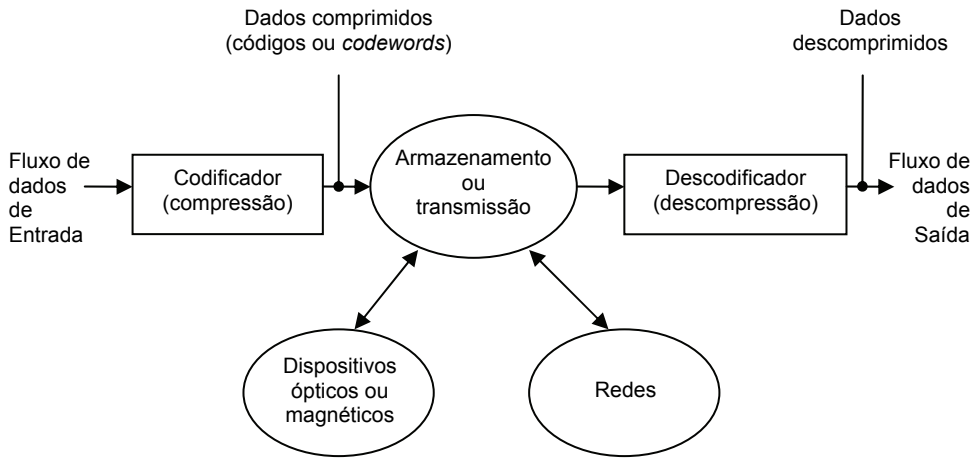

Figura 1.1 - Diagrama de um esquema genérico de compressão

Geralmente, o desempenho de um esquema de compressão pode ser medido, ou referido, em termos do rácio de compressão que proporciona. O rácio de compressão é uma quantidade que se pode obter de várias formas (Salomon, 2004). Contudo, a forma mais comum do rácio de compressão obtém-se dividindo o espaço de armazenamento ocupado pelo fluxo de dados originais pelo espaço consumido pelo fluxo de dados comprimidos ou de códigos (Bhaskaran & Konstantinides, 1997):

$$\text{Rácio de compressão} = \frac{\text{B entrada}}{\text{B saída}}$$

Sendo "B entrada" o número de bits correspondente ao fluxo de dados originais (antes da compressão), e "B saída" o número de bits correspondente ao fluxo de dados comprimidos (códigos), é possível verificar que quanto maior for o valor do rácio de compressão, menor será o comprimento do fluxo de dados comprimidos, isto é, menor será o espaço de armazenamento consumido pelo fluxo de dados comprimidos, ou menor será a largura de banda necessária para a sua transmissão numa rede de computadores.

1.2 MODOS E CATEGORIAS DE COMPRESSÃO

Quanto ao **modo como comprimem a informação**, as técnicas de compressão podem classificar-se como:

- Técnicas de compressão sem perdas (*lossless*);
- Técnicas de compressão com perdas (*lossy*).

© FCA - Editora de Informática

Na **compressão sem perdas**, também designada por *lossless compression*, *bit-preserving* ou *reversible compression*, a informação é recuperada sem qualquer alteração após o processo de descompressão. Isto é, o fluxo de bits descomprimidos é idêntico ao fluxo de bits original (antes da compressão).

Estas técnicas são utilizadas na compressão de dados textuais e numéricos ou na compressão de programas/aplicações que, como é evidente, não devem sofrer qualquer alteração devido ao processo de compressão. As técnicas de compressão sem perdas também são necessárias em certas aplicações multimédia nas quais a exactidão da informação é um factor essencial como, por exemplo, na utilização de imagens para fins médicos.

Na **compressão com perdas**, também designada por *lossy compression* ou *irreversible compression*, a informação descomprimida é diferente da informação original que entretanto foi comprimida. Este modo de compressão adequa-se à maior parte dos tipos de média não-estruturados tais como o áudio digital, as imagens *bitmap* e o vídeo digital.

No entanto, deve-se notar que, na compressão com perdas, o facto de a informação descomprimida ser diferente da informação original não implica necessariamente que a percepção do utilizador seja diferente. Este aspecto essencial das técnicas de compressão com perdas será aprofundado mais adiante e está relacionado com as características particulares dos sentidos do ser humano.

Quanto à **forma como a informação é encarada** pelo esquema de compressão, as técnicas de compressão de dados podem ser agrupadas em duas grandes categorias:

- Técnicas de codificação de entropia (*entropy encoding*);
- Técnicas de codificação de fonte (*source encoding*).

A Tabela 1.1 ilustra as características principais destas categorias.

CATEGORIA	CARACTERÍSTICA PRINCIPAL	MODO DE COMPRESSÃO
ENTROPY ENCODING	Independência das características do fluxo de dados a comprimir	Compressão sem perdas
SOURCE ENCODING	Toma em consideração a semântica do fluxo de dados a comprimir	Compressão com e sem perdas

Tabela 1.1 - Categorias das técnicas de compressão

É importante notar que não se deve confundir a utilização do conceito "codificação" no contexto da compressão com a codificação efectuada como uma das fases do processo de digitalização (Ribeiro, 2004). De igual modo, é importante notar que as técnicas de *entropy* e *source encoding* não são mutuamente exclusivas. Isto significa que os formatos

e normas de codificação de áudio, imagem e vídeo utilizam geralmente combinações de técnicas das duas categorias de modo a obterem o grau ou rácio de compressão mais elevado possível, sendo este o objectivo principal de qualquer técnica de compressão.

As secções que se seguem analisam com mais pormenor as características associadas às técnicas de compressão pertencentes a cada uma destas categorias.

1.3 TÉCNICAS DE CODIFICAÇÃO DE ENTROPIA

Nesta secção apresenta-se uma visão global e introdutória que ilustra as características principais e o modo de funcionamento das técnicas de codificação de entropia mais comuns.

O conceito de codificação de entropia é um termo genérico que se refere às técnicas de compressão e codificação que **não** têm em conta a natureza da informação a ser comprimida. Por outras palavras, as técnicas de compressão baseadas na entropia tratam todos os dados como **sequências de bits**, sem tentarem optimizar a compressão através do conhecimento do tipo de informação que se está a comprimir. Por isso, dizemos que as técnicas de compressão baseadas na entropia **ignoram a semântica** da informação a comprimir. Para além disso, todas as técnicas de codificação de entropia proporcionam sempre um modo de **compressão sem perdas**.

Por exemplo, considere-se a substituição de uma série de 10 *bytes* sucessivos, todos com o valor zero (0), por um carácter especial seguido pelo número 10, obtendo-se por exemplo <!10>. Este é um exemplo típico de uma técnica de codificação de entropia, já que ao realizar este tipo de operação não se está a assumir nenhum pressuposto em relação ao significado deste conjunto de zeros sucessivos.

As técnicas de codificação de entropia podem ainda ser divididas em três tipos principais:

- Técnicas de supressão de sequências repetitivas;
- Técnicas de codificação estatística;
- Técnicas baseadas em dicionários.

1.3.1 TÉCNICAS DE SUPRESSÃO DE SEQUÊNCIAS REPETITIVAS

A técnica de supressão de sequências repetitivas é o primeiro tipo de métodos que se irá analisar, já que é uma das técnicas de compressão mais simples e mais antigas da Ciência da Computação. Esta técnica baseia-se na produção de códigos de comprimento fixo e funciona em dois passos:

1. Detecção de sequências repetitivas de bits ou *bytes*.

© FCA - Editora de Informática

2. Consequente substituição destas sequências pelo seu número de ocorrências.

Este método pode assumir uma de duas formas possíveis, que serão analisadas a seguir:

- Técnicas de supressão de zeros ou espaços;
- Técnica *Run-Length Encoding* (RLE).

1.3.1.1 SUPRESSÃO DE ZEROS OU ESPAÇOS

Nesta forma, o método de supressão de sequências repetitivas assume que apenas um carácter (*byte*) predeterminado aparece frequentemente e é repetido. Este carácter pode ser o zero em dados numéricos ou o espaço em dados textuais. Consequentemente, uma série de **n** espaços, ou zeros, sucessivos será substituída por um carácter especial (designado por *flag* ou *meta character*) imediatamente seguido pelo número de ocorrências (**n**) desse carácter.

Por exemplo, consideremos que temos a seguinte cadeia de caracteres numéricos em que cada carácter (neste caso, cada algarismo) é codificado com um *byte*: 7420000000000005. Utilizando a técnica de supressão de zeros, e definindo o carácter "!" como a *flag*, é possível obter a seguinte sequência comprimida: 742**!12**5, já que existe uma sequência de 12 zeros consecutivos no fluxo de dados original.

No contexto deste exemplo é pertinente realizar duas observações. Em primeiro lugar note-se que o número 12 que segue a *flag* deverá ser codificado com um *byte*, de modo a evitar qualquer confusão com o carácter 5 que se segue. Em segundo lugar, notando que se partiu de uma sequência com 16 *bytes* e se obteve uma sequência comprimida com o comprimento de 6 *bytes*, é possível determinar o rácio de compressão obtido do seguinte modo:

$$\text{Rácio de compressão} = \frac{16\,bytes}{6\,bytes} = 2{,}66$$

Isto significa que se conseguiu comprimir 2,66 unidades de informação do fluxo de dados original numa única unidade de informação no fluxo de dados comprimidos.

1.3.1.2 *RUN-LENGTH ENCODING*

Na forma RLE, o método de supressão de sequência repetitivas permite que qualquer sequência de caracteres repetidos possa ser substituída por uma forma abreviada. Isto significa que a técnica RLE permite comprimir qualquer carácter que surja de forma repetitiva, para além dos zeros e espaços.

© FCA - Editora de Informática

O algoritmo da técnica RLE consiste em substituir uma série de **n** caracteres "**c**" consecutivos pelo próprio carácter "**c**" precedido por um carácter especial (a *flag* ou *escape character*) que, por sua vez, é seguido pelo número **n** de ocorrências do carácter repetido (Fluckiger, 1995). Este conjunto de 3 caracteres que substitui a sequência repetida designa-se por *token*, e representa-se do seguinte modo: **!<n><c>**.

Analisando o modo de funcionamento descrito, é possível concluir que este método **não** deve ser utilizado nos casos em que um carácter surge repetido apenas duas vezes visto que originaria uma sequência mais comprida do que a sequência original. De igual modo, a sua utilização para substituir sequências de três caracteres sucessivos não traria qualquer vantagem. Assim, pode concluir-se que a substituição deve apenas ser realizada caso o número de ocorrências sucessivas de um carácter seja igual ou superior a quatro.

A forma do método RLE que acabou de se descrever é a forma mais simples e utiliza-se apenas para conteúdos textuais. Existem, contudo, outras variações deste método que podem ser consultadas, por exemplo, na bibliografia aconselhada (Nelson & Gailly, 1995; Salomon, 2004; Li & Drew, 2004).

Para exemplificar o funcionamento da técnica RLE, consideremos a seguinte cadeia de caracteres: ABCCCCCCABCABC. Ora, o método RLE substitui sequências de dados redundantes por *tokens*, pelo que utilizando a codificação RLE é possível comprimir a sequência repetitiva de "C" utilizando uma determinada *flag* tal como o carácter "!". Assim, a forma comprimida da cadeia de caracteres original poderia assumir a seguinte expressão: AB**!6C**ABCABC.

Neste caso, o rácio de compressão obtido pode ser determinado do seguinte modo:

$$\text{Rácio de compressão} = \frac{14\,bytes}{11\,bytes} = 1{,}27$$

Este exemplo permite ilustrar o funcionamento do algoritmo de compressão RLE. Inicia-se a leitura do fluxo de caracteres e, após a leitura do primeiro carácter, actualiza-se o valor do contador para 1, armazenando-se o carácter num *buffer* de memória. Os caracteres que se seguem são então comparados com o carácter que está guardado no *buffer* e, caso sejam idênticos, incrementa-se novamente o contador. Logo que se efectue a leitura de um carácter diferente, a operação a efectuar depende do valor armazenado no contador. Se for inferior a 3, os caracteres são escritos no fluxo de dados comprimidos, e o novo carácter é agora armazenado no *buffer* para comparação. Caso contrário, escreve-se a *flag* no fluxo de dados comprimidos, seguida do valor do contador e do carácter presente no *buffer*.

No que diz respeito à descompressão, sempre que se encontra a *flag* no fluxo de dados comprimidos, efectua-se uma operação de leitura do valor **n** e do carácter **C** que se

© FCA - Editora de Informática

seguem, escrevendo-se o carácter **C** no fluxo de dados descomprimidos tantas vezes quantas o valor de **n**.

Para além de dados textuais, a técnica RLE pode igualmente ser aplicada a dados de imagem (Salomon, 2004). Como é sabido, uma imagem *bitmap* é constituída por um conjunto de píxeis, ocupando cada píxel 1 bit caso se trate de uma imagem a preto e branco, ou mais do que 1 bit se a imagem for colorida (por exemplo, uma imagem com 4 bpp (bits por píxel) pode conter até 16 cores diferentes). Neste caso, o fluxo de dados de entrada é constituído pelos bits armazenados em cada um dos píxeis da imagem.

Normalmente, o primeiro píxel da imagem é o que se situa no canto superior esquerdo, ao passo que o último se situa no canto inferior direito. Partindo do primeiro píxel, o codificador RLE lê os restantes píxeis ordenadamente, linha a linha, até se atingir o último píxel, em busca de grupos repetitivos de píxeis (*runs*). É, contudo, possível efectuar o varrimento dos píxeis da imagem seguindo um esquema diferente, por exemplo, coluna a coluna, ou de acordo com uma ordenação ziguezague (descrita no Capítulo 5 no âmbito da análise da norma JPEG).

A compressão RLE aplicada ao caso das imagens baseia-se no seguinte pressuposto: se um píxel da imagem tem um determinado valor, é muito provável que os píxeis vizinhos assumam igualmente o mesmo valor, isto é, possuam a mesma cor do píxel que consideramos inicialmente.

Como se pode facilmente deduzir, este método funciona bem (proporciona um bom rácio de compressão) apenas nos casos em que as imagens contêm pouco detalhe, e os valores dos píxeis não variam muito (por exemplo, no caso em que as imagens resultam do *rendering* de objectos gráficos desenvolvidos no computador). Por isso, quanto maior for a complexidade da imagem (por exemplo, no caso de imagens que resultam de fotografias digitais ou digitalizadas) mais baixo será o rácio de compressão, e portanto a eficiência do método RLE.

Para ilustrar o processo de compressão RLE de uma imagem colorida com 8 bpp, considere-se, por exemplo, a primeira linha da imagem contendo os seguintes valores:

75,75,75,75,75,75,10,45,58,30,30,30,30,8,8,8,8,8,3,…

Aplicando o método RLE, e assinalando a sublinhado os valores do número de ocorrências das *runs*, o resultado da compressão da primeira linha desta imagem seria:

<u>6</u>,75,10,45,58,<u>4</u>,30,<u>5</u>,8,3,…

Como se pode verificar, o único problema que existe é: para cada *byte*, distinguir se o seu valor corresponde a um valor de cor do píxel (como 75) ou a um valor do número de ocorrências (como <u>6</u>).

© FCA - Editora de Informática

Salomon (2005) sugere algumas soluções para este problema, incluindo as seguintes:

- Caso a imagem contenha apenas 128 valores diferentes de tonalidades de cinzento (*grayscales*), é possível dedicar 1 bit de cada *byte* para assinalar se o *byte* contém um valor de cor ou um número de ocorrências;

- Se a imagem contiver 256 tonalidades diferentes de cinzento (0 a 255), poderá ser reduzida para 255 tonalidades, reservando um dos 256 valores (por exemplo, o último valor de 255) para ser utilizado como *flag*. Neste caso, a sequência comprimida do exemplo acima ficaria:

255,6,75,10,45,58,255,4,30,255,5,8,3,…

1.3.2 TÉCNICAS DE CODIFICAÇÃO ESTATÍSTICA

A codificação estatística, também designada por *statistical encoding*, é o segundo tipo de técnicas de codificação de entropia que se irá analisar. A codificação estatística é igualmente um tipo de técnicas bastante utilizado em formatos de codificação multimédia, apesar de ser menos rudimentar do que o método de supressão de sequências repetitivas, proporcionando técnicas em que a compressão é simétrica, isto é, o codificador e o descodificador possuem uma complexidade idêntica.

O método de codificação estatística baseia-se no seguinte processo genérico:

1. Identificação dos padrões de bits ou *bytes* que ocorrem mais frequentemente num dado fluxo de dados.
2. Codificação de cada padrão com menos bits do que o número de bits dispendido para o representar no fluxo de dados original.

De acordo com Salomon (2004), na codificação estatística os padrões que ocorrem com **menor frequência** serão codificados mediante a utilização de **mais** bits, ao passo que os padrões **mais frequentes** serão substituídos por códigos **menos** extensos. A noção fundamental assenta, pois, na seguinte observação: na codificação estatística, os padrões de bits ou *bytes* são substituídos de acordo com a frequência com que ocorrem, sendo este o motivo porque se designa por estatística. Os padrões mais frequentes utilizam códigos mais curtos, ao passo que os padrões menos frequentes utilizam códigos mais extensos.

No entanto, já que o método da codificação estatística implica a substituição de padrões por outros padrões menos ou mais extensos, segue-se que deve existir uma tabela, normalmente designada por *codebook* ou tabela de códigos, que estabeleça a correspondência entre os padrões originais e o seu novo código, quer no lado da codificação quer no da descodificação. Em certos casos, a identificação da frequência do padrão e a atribuição de códigos já se encontra disponível ou é predefinida. Neste caso, a frequência de ocorrência dos padrões não necessita de ser determinada de cada vez que se codifica um novo fluxo

© FCA - Editora de Informática