TEORÍA DE LAS COMUNICACIONES

TRABAJO PRÁCTICO DE SIMULACIÓN

PARTE 2: Codificación de fuente y canal

Ivan Svetlich

06/08/2021

Introducción

En el presente informe se estudia la implementación de codificación de canal y de fuente en un sistema de comunicación, con la asistencia de simulaciones realizadas en Matlab. En la Parte 1 se analizan las capacidades de un código lineal de bloque aplicado como detector y corrector de errores. Los resultados obtendios se comparan con las expresiones teóricas y con un sistema sin codificación. En la Parte 2 se utiliza el algoritmo de codificación de Huffman con fuentes extendidas para comprimir una imagen binaria y se verifica la reducción del tamaño del archivo.

1 Codificación de canal

Códigos de bloque lineales

Un código de bloque lineal (n, k) puede representarse con una matriz generadora G, que mapea las 2^k palabras de fuente posibles a sus correspondientes palabras de código de longitud n. Si la matriz G tiene la forma

$$G = [I_k|P] \tag{1}$$

donde I_k es una matriz identidad de $k \times k$ y P es una matriz de $k \times (n-k)$, entonces el código resultante se denomina sistemático. En este tipo de códigos los primeros k elementos de la palabra de código son la secuencia del mensaje, y los siguientes $n-k$ son los bits de paridad. Esta característica otorga una

facilidad al momento de la decodificación, pues basta con seleccionar los primeros k elementos para recuperar los bits de fuente.

En este caso se requiere diseñar un código de bloque lineal $(9, 5)$ con codificación sistemática, así que su matriz generadora será

$$G = [I_5|P] \tag{2}$$

La matriz de paridad propuesta es

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

por lo que la matriz generadora resulta

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

Dado un código C asociado a una matriz G, su complemento ortogonal define un código $C^{\perp}$ asociado a una matriz generadora H, cuyas filas son ortogonales a las de G. Como toda las palabra c perteneciente a C es ortogonal a las filas de H

$$cH^t = 0 \tag{3}$$

Esta propiedad permite controlar si una palabra pertenece al conjunto de palabras de código de C, y por lo tanto H se denomina matriz de control de paridad. En el caso de los códigos sistemáticos H esta dada por

$$H = \left[P^t|I_{n-k}\right] \tag{4}$$

donde P^t denota la matriz de paridad P transpuesta. Para el código propuesto, la matriz de control de paridad resulta

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

Las capacidades de detección y corrección de errores de un código de bloque lineal están determinadas por la distancia mínima entre dos palabras de código d_{min}, que en este tipo de códigos coincide con el peso mínmo w_{min}. Para que se produzca un fallo en la detección de errores, estos se deben dar de forma tal que la secuencia recibida sea una palabra de código. Esto puede ocurrir solo si la cantidad de errores es mayor o igual a d_{min}. Entonces, la capacidad de detección es

$$t_d = d_{min} - 1 \tag{5}$$

De forma similar, un fallo en la correción de errores puede ocurrir cuando la cantidad de errores en una secuencia sea tal que la palabra de código más cercana ya no sea la transmitida, sino otra de las palabras de código posibles. Entonces, la capacidad de correción es

$$t_c = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor \tag{6}$$

Mediante simulación se halló que la distancia mínima del código propuesto es $d_{min} = 3$. Utilizando las ecuaciones (5) y (6)

- $t_d = 2$
- $t_c = 1$

La probabilidad de error de palabra de un sistema con codificación mediante un código de bloque lineal (n, k) corrector de t_c errores, utilizando decisión dura, es

$$P_{ep} = \sum_{i=t_c+1}^{n} \binom{n}{i} p^i \left(1-p\right)^{n-i} \tag{7}$$

donde p es la probabilidad de error en el canal. Para realizar una comparación apropiada con un sistema sin codificación, se debe tener en cuenta la distribución de la energía disponible en ambos casos. Por ejemplo, si se transmite sin codificación con modulación BPSK, se tiene

$$P_{eb} = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) \tag{8}$$

Entonces, la probabilidad de error de bit en el canal cuando se utiliza un código (n, k) es

$$p = Q\left(\sqrt{\frac{2E_b}{N_0}\, k/n}\right) \tag{9}$$

A partir de (7) se puede obtener el valor aproximado de la probabilidad de error de bit

$$P_{eb} \sum_{i=t_c+1}^{n} \frac{i}{n} \binom{n}{i} p^i \left(1-p\right)^{n-i} \tag{10}$$

Con el mismo razonamiento pueden hallarse las expresiones de probabilidad de error cuando el código se utiliza como detector. Una aproximación más ajustada se obtiene si, en lugar de considerar como errores todas las palabras con un peso de Hamming mayor a $t_d + 1$, solo se tienen en cuenta las que cumplen esta condición y no son palabras de código. En la tabla 1 se listan las expresiones correspondientes a transmisión BPSK sin codificación, codificación con detección y codificación con corrección, donde p se calcula a partir de la expresión (9).

Tabla 1: Expresiones de probabilidad de error

	Sin codificación	**Código detector**	**Código corrector**
$\mathbf{P_{ep}}$	$1 - \left(1 - Q\left(\sqrt{\frac{2E_b}{N_0}}\right)\right)^k$	$\sum_{i=t_d+1}^{n} \binom{n}{i} p^i \left(1-p\right)^{n-i}$	$\sum_{i=t_c+1}^{n} \binom{n}{i} p^i \left(1-p\right)^{n-i}$
$\mathbf{P_{eb}}$	$Q\left(\sqrt{\frac{2E_b}{N_0}}\right)$	$\sum_{i=t_d+1}^{n} \frac{i}{n} \binom{n}{i} p^i \left(1-p\right)^{n-i}$	$\sum_{i=t_c+1}^{n} \binom{n}{i} p^i \left(1-p\right)^{n-i}$

El objetivo fundamental de la codificación es reducir la probabilidad de error de bit de fuente respecto de una comunicación sin codificación para una dada energía invertida, o bien reducir la energía necesaria para lograr una determinada probabilidad de error. La ganancia de un código se mide como

$$G_c = \frac{\frac{E_b}{N_0}\ con\ cod.}{\frac{E_b}{N_0}\ sin\ cod.}^{@P_{eb}} \tag{11}$$

La ganancia asintótica cuando $P_{eb} \to 0$ o $E_b/N_0 \to \infty$ determina una cota máxima para la ganancia de un código. En el caso de decisión dura se puede hallar como

$$G_a = \frac{k}{n} \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor \tag{12}$$

Para el código propuesto su valor es

$$G_a = \frac{5}{9} \left\lfloor \frac{3+1}{2} \right\rfloor = \frac{10}{9} = 0.458\ dB \tag{13}$$

En la figura 1 se muestra la curva de probabilidad de error de palabra cuando se utiliza el código como corrector de errores (en azul). A modo comparativo se incluyen las curvas de probabilidad de error obtenidas a partir de las expresiones teóricas para los casos con codificación (en rojo) y sin codificación (en verde). Se observa que a bajas E_b/N_0 la probabilidad de error con correción es mayor que sin codificación, debido al aumento en la probabilidad de que surjan patrones de error que son corregidos de forma desacertada.

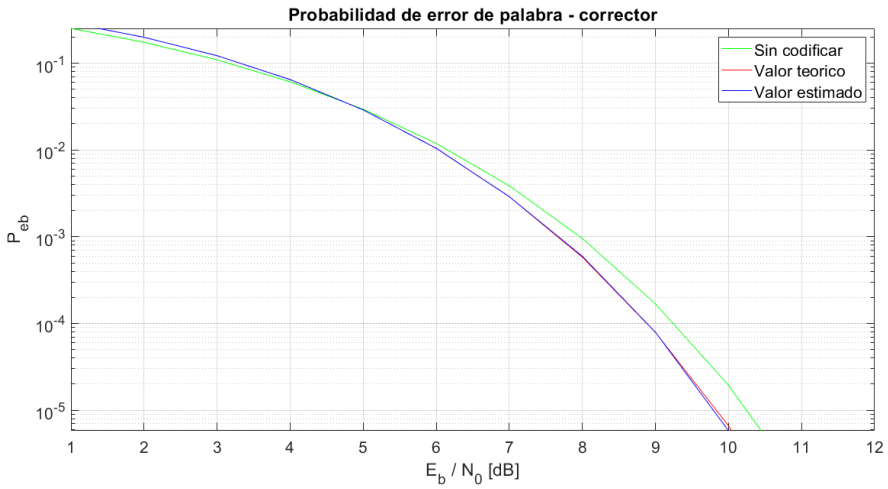

Figura 1: Probabilidad de error de palabra (corrector).

Comparando las curvas de probabilidad de error de bit de fuente en las mismas condiciones que en el caso anterior (figura 2), se observa que la codificación supone una ganancia respecto de una comunicación sin codificación a partir de 6 dB, aproximadamente. Por debajo de este valor de E_b/N_0 no resulta conveniente utilizar el código como corrector. A medida que $P_{eb} \to 0$ aumenta la proporción de patrones de error que el código es capaz de corregir adecuadamente, y la ganancia por la codificación las curvas tiende al valor de ganancia asintótica del código.

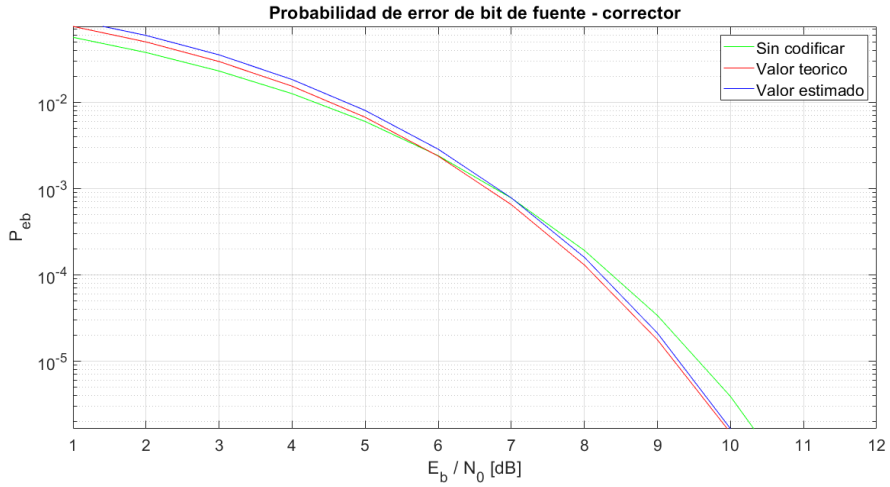

Figura 2: Probabilidad de error de bit de fuente (corrector).

De forma similar se relevan la curvas de probabilidad de error de palabra y de bit de fuente cuando se utiliza el código como detector de errores (figuras 3 y 4). Evidentemente, la aproximación teórica utilizada es pesimista y resulta en una cota holgada para el valor obtenido mediante simulación. A diferencia del caso anterior, el código detector ofrece un mejor desempeño en todo el rango estudiado. Esto se debe a que el código detector planteado descarta cualquier palabra cuyo síndrome sea distinto de 0, y los errores cometidos corresponden a patrones de errores que no pueden ser detectados, por lo que su desempeño siempre será mejor que sin codificación.

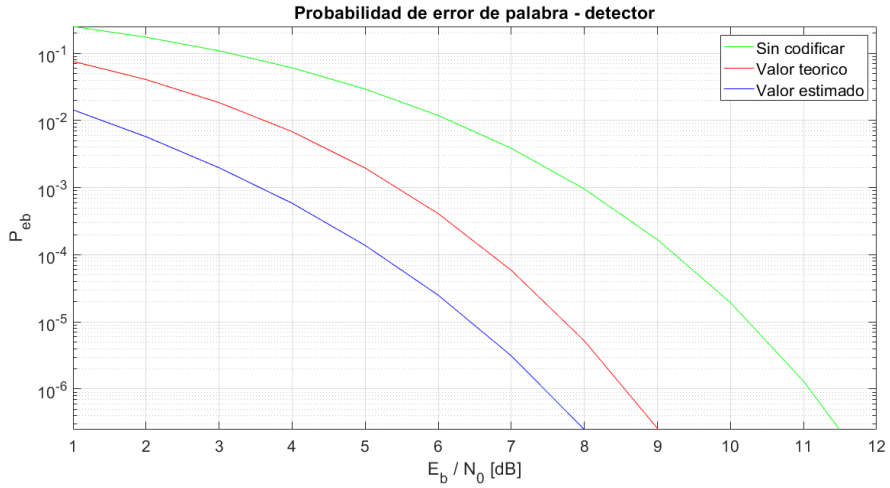

Figura 3: Probabilidad de error de bit de palabra (detector).

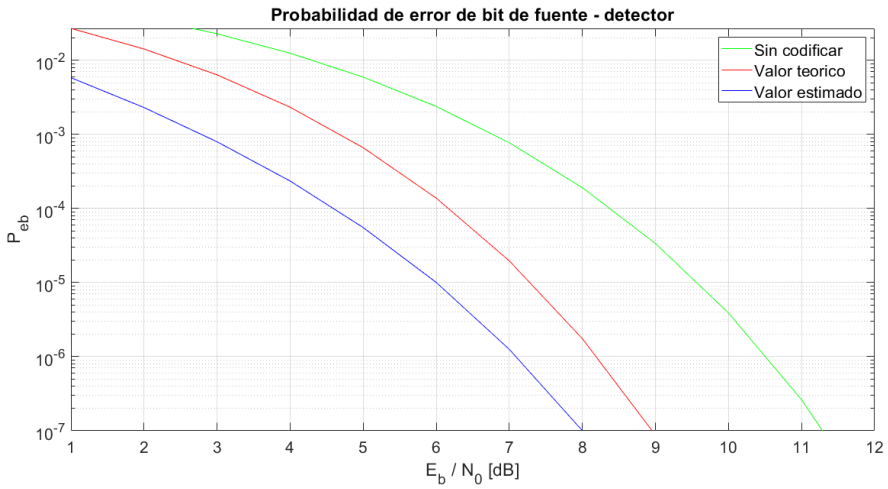

Figura 4: Probabilidad de error de bit de fuente (detector).

2 Codificación de fuente

Algoritmo de codificación de Huffman

El objetivo de los algoritmos de compresión de datos es representar a una fuente con el menor número de bits que permita su recuperación por parte del receptor. Dependiendo de la tolarancia frente a la distorsión de la información, la compresión puede ser con o sin pérdidas. En una compresión sin pérdidas, la fuente se comprime de manera tal que la reconstrucción de la información es perfecta. En cambio, una compresión con pérdidas implica una distorsión de la fuente dentro de un límite tolerado.

El algoritmo de codificación de Huffman es uno de los principales métodos de compresión sin pérdidas de fuentes discretas. Se trata de un algoritmo óptimo, en el sentido de que el largo promedio de las palabras de código es el mínimo posible. Este parámetro se define como

$$\bar{L} = \sum_{k=0}^{K-1} p_k l_k \ [bits/msj] \tag{14}$$

donde p_k es la probabilidad del k-ésimo mensaje, l_k su longitud y K el número total de mensajes posibles. Además, como las palabras de código están sujetas a la condición de prefijo (ninguna palabra es un prefijo de otra de mayor longitud), la secuencia recibida es unívoca e instantáneamente decodificable.

Según el teorema de codificación de fuente de Shannon, el mínimo largo de palabra promedio posible está determinado por la entropía de la fuente,

si el código es unívoco. La entropía es una medida de la incertidumbre o ambiguedad de la fuente y esta dada por

$$H\left(S\right) = -\sum_{k=0}^{K-1} p_k \log_2\left(p_k\right) \ [bits/msj] \tag{15}$$

A partir de esta definición puede hallarse la eficiencia del código como

$$\eta = \frac{H(S)}{\bar{L}} \tag{16}$$

Si para un código de prefijo se tiene que $\bar{L} > H(S)$, este puede mejorarse utilizando una fuente extendida S^n, que se obtiene tomando como mensajes la combinación de n mensajes de la fuente original.

En este caso se requiere diseñar el algoritmo de Huffman para comprimir un archivo de dimensiones $434 \times 432\, bits$, utilizando fuentes extendidas de orden 2 y 3. Para estimar las probabilidades de cada uno de los mensajes posibles m_i, se calculan las frecuencias relativas dentro del archivo

$$f_i = \frac{n^\circ\, de\, ocurrencias\, de\, m_i}{longitud\, total\, en\, bits} \times orden\, de\, la\, fuente\, extendida \tag{17}$$

Los resultados obtenidos mediante simulación se muestran en las tablas 1 y 2.

Tabla 2: Frecuencias relativas (orden 2)

m_i	m_0	m_1	m_2	m_3
mensaje	0 0	0 1	1 0	1 1
frecuencia	0.47646	0.0090886	0.0087686	0.50569

Tabla 3: Frecuencias relativas (orden 3)

m_i	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7
mensaje	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
frecuencia	0.46941	0.00837	0.00008	0.00808	0.00752	0.00001	0.00790	0.49862

A partir de los valores hallados, se obtienen las palabras de código correspondientes a cada mensaje, siguiendo el criterio de que a los mensajes menos probables se les asigna el bit 1. En la tabla 3 se muestran las probabilidades resultantes del procedimiento para fuente de orden 2, y en la tabla 4 las palabras de código correspondientes a cada mensaje.

Tabla 4: Fuente extendida de orden 2

1	**2**	**3**	**4**
$p_3 = 0.50569$	$p_3 = 0.47646$	$p_3 = 0.50569$	$p_{0,1,2,3} = 1$
$p_0 = 0.47646$	$p_0 = 0.47646$	$p_{0,1,2} = 0.494317$	
$p_1 = 0.0090886$	$p_{1,2} = 0.017857$		
$p_2 = 0.0087686$			

Tabla 5: Palabras de código (orden 2)

Mensaje	**Palabra de código**	**Longitud**
$m_0 = 0\,0$	1 0	2
$m_1 = 0\,1$	1 1 0	3
$m_2 = 1\,0$	1 1 1	3
$m_3 = 1\,1$	0	1

Con la misma metodología se hallan las palabras de código de la fuente extendida de orden 3. El cálculo de probabilidades se muestra en la tabla 5 y el diccionario resultante en la tabla 6.

Tabla 6: Cálculo de fuente extendida de orden 3

1	2	3	4	5	6	7	8
$p_7 = 0.49862$	$p_7 = 0.49862$	$p_7 = 0.49862$	$p_7 = 0.49862$	$p_7 = 0.49862$	$p_7 = 0.49862$	$p_{0,1,2,3,4,5,6} = 0.50137$	$p_T = 1$
$p_0 = 0.46941$	$p_0 = 0.46941$	$p_0 = 0.46941$	$p_0 = 0.46941$	$p_0 = 0.46941$	$p_0 = 0.46941$	$p_7 = 0.49862$	
$p_1 = 0.00837$	$p_1 = 0.00837$	$p_1 = 0.00837$	$p_{2,4,5,6} = 0.01552$	$p_{1,3} = 0.01644$	$p_{1,2,3,4,5,6} = 0.03197$		
$p_3 = 0.00808$	$p_3 = 0.00808$	$p_3 = 0.00808$	$p_1 = 0.00837$	$p_{2,4,5,6} = 0.01552$			
$p_6 = 0.00790$	$p_6 = 0.00790$	$p_6 = 0.00790$	$p_3 = 0.00808$				
$p_4 = 0.00752$	$p_4 = 0.00752$	$p_{2,4,5} = 0.00762$					
$p_2 = 0.00008$	$p_{2,5} = 0.00010$						
$p_5 = 0.00001$							

Tabla 7: Palabras de código (orden 3)

Mensaje	**Palabra de código**	**Longitud**
$m_0 = 0\,0\,0$	$0\,0$	2
$m_1 = 0\,0\,1$	$0\,1\,0\,0$	4
$m_2 = 0\,1\,0$	$0\,1\,1\,1\,1\,0$	6
$m_3 = 0\,1\,1$	$0\,1\,0\,1$	4
$m_4 = 0\,1\,0$	$0\,1\,1\,1\,0$	5
$m_5 = 0\,1\,1$	$0\,1\,1\,1\,1\,1$	6
$m_6 = 1\,1\,0$	$0\,1\,1\,0$	4
$m_7 = 1\,1\,1$	1	1

La tasa de compresión es un parámetro que cuantifica la reducción en la longitud obtenida al utlizar codificación

$$R_c = \frac{\text{largo sin compresión}}{\text{largo con compresión}} \tag{18}$$

donde un valor mayor a 1 indica que se logró compresión. En la tabla 7 se presentan los resultados obtenidos al evaluar el desempeño de los códigos

diseñados. A modo comparativo, se incluyen los datos de desempeño correspondientes a una fuente de orden 1, que en este caso no logra compresión por tratarse de una fuente con mensajes (píxeles) prácticamente equiprobables.

Tabla 8: Desempeño de los códigos diseñados

Orden	$H\left(\boldsymbol{S}^n\right)$	$\bar{L}$	η	R_c
1	0.99938	1	0.99938	1
2	1.99877	1.51217	1.32178	1.32260
3	2.99815	1.57303	1.90597	1.90715

Aunque los dos mensajes posibles en el archivo (pixel blanco y pixel negro) son prácticamente equiprobables, la utilización de fuentes extendidas logra compresión gracias a los patrones presentes en la imagen. Al haber zonas amplias de un mismo color, la frecuencia de los mensajes correspondientes a una sucesión de símbolos iguales es mucho mayor a la de las transiciones. Como la codificación de Huffman asigna palabras de código de menor longitud a los mensajes más probables, esto se traduce en una reducción del tamaño total del archivo.

Automatizando el proceso de codificación para una fuente de orden n, se obtuvo el largo promedio de palabra de código para un rango de valores de n. La curva resultante se muestra en la figura 5.

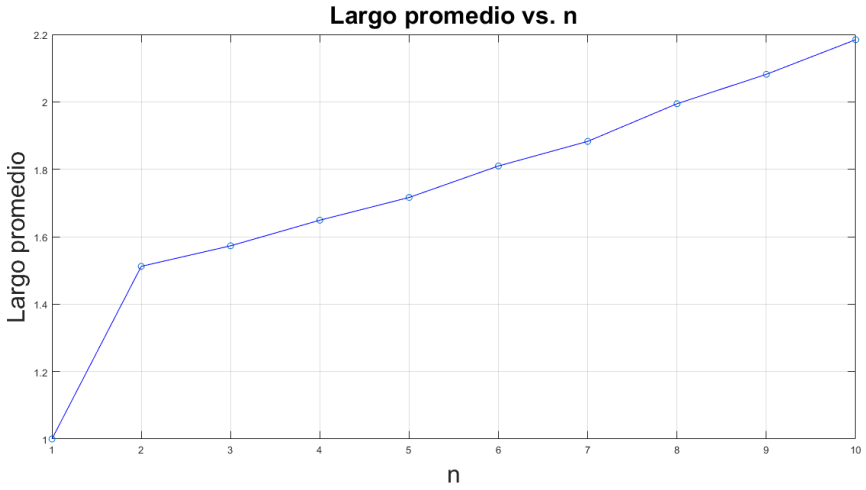

Figura 5: Largo promedio vs. n.

Referencias

[1] S. Haykin. *Communications Systems, 4th Edition.* John Wiley & Sons, 2001.

[2] J. Proakis and M. Saleh. *Digital Communications, 5th Edition.* McGraw Hill, 2007.