Construcción de códigos ldpc con conjuntos de parada de gran tamaño

Salazar Ripoll[1], Juan Camilo[2] y Barraza, Néstor Rubén[3]

[1]Juan Camilo Salazar Ripoll. Universidad de los Andes. Estudiante de la maestría en Ingeniería Matemática de la Universidad de Buenos Aires. **E mail:** palazar43@gmail.com
[3]Néstor Rubén Barraza. Profesor Asociado. Universidad de Tres de Febrero. Profesor Adjunto. Facultad de Ingeniería. Universidad de Buenos Aires. **E mail:** nestor.barraza@gmail.com

Resumen

En este trabajo se presenta un nuevo algoritmo para construir códigos LDPC con una distancia de parada grande. Dado que el rendimiento del código es caracterizado por el tamaño del mínimo conjunto de parada, resulta importante encontrar los conjuntos de parada en los códigos LDPC. Si el tamaño mínimo de los conjuntos de parada es grande, se evita que el código quede atrapado en ciclos, especialmente en los canales binarios de borrado. Encontrar el conjunto de parada mínimo no es una tarea fácil, ya que es un problema NP Completo. Consecuentemente, se propone la construcción de un código LDPC con un tamaño de parada grande determinado al momento de su construcción. El rendimiento del código obtenido de esta manera es analizado mediante simulaciones.

Palabras clave: LDPC, conjunto de parada, BEC, algoritmo

Abstract

A new algorithm to construct good LDPC codes with large stopping sets is presented. Since the minimum stopping set characterizes an LDPC code, searching for stopping sets in LDPC codes is an important issue. Large minimum stopping sets avoid the LDPC code to get trapped in cycles specially on the binary erasure channel. Dealing with stopping sets is not an easy task since their discovering is a well known NP hard problem. Conversely, we propose an algorithm in order to construct an LDPC code from a stopping set which is demonstrated to be large. Results of simulations showing the performance of the LDPC code obtained this way are analyzed.

Keywords: LDPC, stopping set, BEC, algorithm

Introducción

Los códigos LDPC han sido una importante rama de estudio en Teoría de Información y Codificación desde que fueron redescubiertos por MacKay y Neal en [1], muchos años más tarde de que fueran introducidos por Gallager en [2]. El avance tecnológico en procesadores y memorias de los últimos años permitió el desarrollo de algoritmos de decodificación de baja complejidad como propagación de la verosimilitud o de suma-producto. En el presente, tanto los códigos LDPC como los turbocódigos son dos importantes ramas en teoría de codificación con los cuales se obtiene un rendimiento cercano a la capacidad del canal. Para obtener códigos LDPC de buen rendimiento se deben tener en cuenta varias características, una de ellas es el tamaño del conjunto de parada, a mayor tamaño, mejor rendimiento, ya que conjuntos de parada de gran tamaño evitan que el algoritmo quede atrapado en ciclos y de este modo, se asegura la convergencia, principalmente, cuando se estudia el canal de borrado. Los métodos utilizados usualmente para analizar el rendimiento de códigos LDPC consisten en buscar los conjuntos de paradas existentes, ver por ejemplo [3] y [4]. Este método resulta de difícil aplicación, ya que la búsqueda de conjuntos de parada es un problema NP completo, ver [5]. También se han desarrollado algunos algoritmos para construir códigos LDPC con conjuntos de parada grandes, [6]. Otras técnicas están basadas en grafos coset, [7], y en redes multidimensionales finitas, [8]. En este trabajo, proponemos un nuevo algoritmo para construir códigos LDPC expan-diendo un grafo inicial aumentando el grado de los nodos de chequeo para una cintura dada. Este artículo está organizado como sigue: en la sección 2 se hace una descripción de los códigos LDPC. En la sección 3 se hace una descripción detallada del algoritmo. En la sección 4, se analiza el rendimiento del algoritmo a través de un BEC. Se presentan algunas conclusiones en la sección 5.

Códigos LDPC

Los códigos LDPC son un caso particular de códigos de bloque con la particularidad de que la matriz de chequeo de paridad H tiene la mayoría de los elementos nulos, de ahí el nombre de baja densidad. Esto permite tener una matriz grande, es decir que el tamaño de bloque del código es grande, reduciendo de esta manera la probabilidad de error del bloque. Esto último es consistente con el teorema de Shannon que demuestra que la probabilidad de error de un código con una tasa menor que la capacidad del canal tiende a cero cuando el tamaño del bloque tiende a infinito. Para una explicación más detallada de los códigos LDPC puede verse [9].

Representación de códigos LDPC

La decodificación de códigos LDPC es una materia de estudio constante. Existen actualmente algoritmos muy eficientes para decodificar códigos LDPC basados en envío de mensajes, estos algoritmos eran imposibles de implementar en la época de Gallager. Una explicación de estos algoritmos puede verse en [9].

En un código de bloque existen ecuaciones de chequeo de paridad que conectan los bits del mensaje transmitido, por ejemplo en el siguiente código (7,2) donde el mensaje transmitido tiene 7 bits y 5 son de redundancia la matriz de chequeo de paridad es la siguiente:

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

y las ecuaciones de chequeo son las siguientes:

$$\begin{aligned} x_1 + x_3 + x_5 &= 0 \\ x_2 + x_4 + x_6 &= 0 \\ x_4 + x_6 + x_7 &= 0 \\ x_1 + x_3 + x_6 &= 0 \\ x_2 + x_5 + x_7 &= 0 \end{aligned}$$

Las ecuaciones de paridad pueden representarse en un gráfico con dos tipo de nodos, nodos variable y nodos de chequeo. Los nodos variable corresponden a los bits del mensaje, y los nodos de chequeo corresponden a las ecuaciones de paridad. Un nodo variable se conecta a un nodo de chequeo que corresponde a la ecuación de paridad en la que interviene. Esto determina conexiones entre los nodos variable y los nodos de chequeo. Por ejemplo, el código mencionado anteriormente se ve representado en la figura 1.

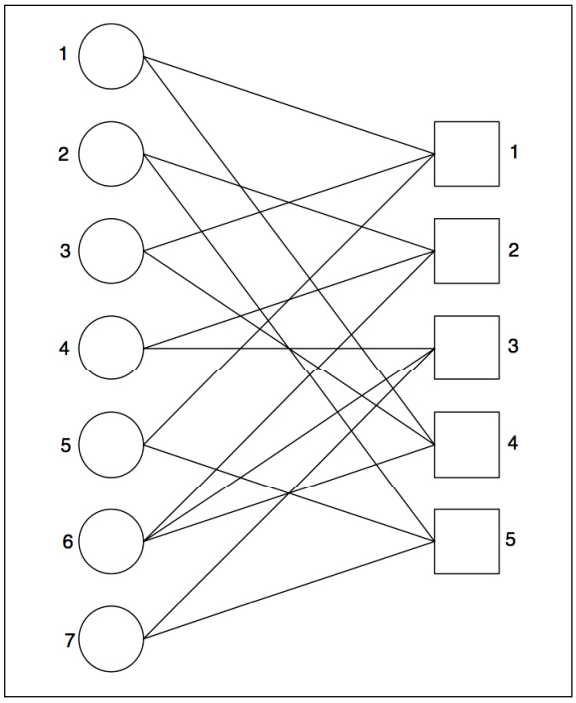

Figura 1. Grafo de Tanner del código (7,2)

Este gráfico se conoce como grafo de Tanner, tiene la propiedad de que las conexiones se establecen entre el conjunto de nodos variable y el conjunto de nodos de chequeo. Un grafo con esta característica se conoce como grafo bipartito.

Los algoritmos para decodificar códigos LDPC se basan en propagación de la verosimilitud o pase de mensajes. El algoritmo de pase de mensajes consiste en información que es enviada entre los nodos variables que corresponden a los bits transmitidos y los nodos de chequeo que corresponden a las ecuaciones de paridad.

Canal de borrado binario

Un canal binario es aquel sobre el cual se pueden transmitir solamente dos símbolos, en este caso 1 ó 0. Un canal de borrado binario es aquel que al transmitir un bit (1 o 0), el bit recibido es "borrado" con una probabilidad

Este tipo de canal sugerido inicialmente en 1954 por Elias, encuentra actualmente aplicaciones en internet, donde los paquetes de datos pueden arribar correctamente o perderse debido a superar la capacidad de procesamiento o demasiado retardo.

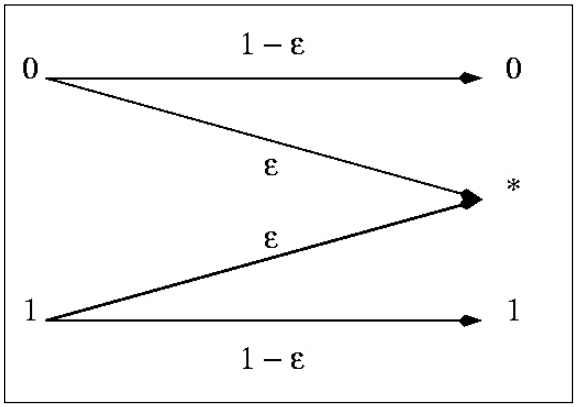

Figura 2. Transmisión de un bit a través de un canal binario de borrado

Conjuntos de parada

Sea X el conjunto de los nodos variable y sea Y el conjunto de los nodos de chequeo. Se dice que un subconjunto P de X es un conjunto de parada si los vecinos de P están conectados a P al menos 2 veces. En otras palabras, que haya al menos 2 ecuaciones que contengan cada uno de los nodos variable en P.

Si se considera la matriz cuyas columnas son un subconjunto de las columnas de H, esta matriz representa un conjunto de parada si las filas que no son nulas tienen al menos dos 1's.

En el código del ejemplo anterior, un conjunto de parada es aquel subconjunto formado por los nodos 2, 4 y 7, como se puede ver en la siguiente submatriz obtenida con las columnas 2, 4 y 7 de la matriz H:

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

El grafo de Tanner con las conexiones de los nodos dentro del conjunto de parada se muestra en la figura 3.

Cuanto mayor es el conjunto de parada, mejor es el código, ya que permite corregir un mayor número de errores y es menos probable que quede atrapado en ciclos. De ahí la importancia de que los códigos LDPC más eficientes sean los que tengan conjuntos de parada grandes.

Algoritmo de construcción

Construcción a partir de grafos simples

Se va a presentar un método para construir

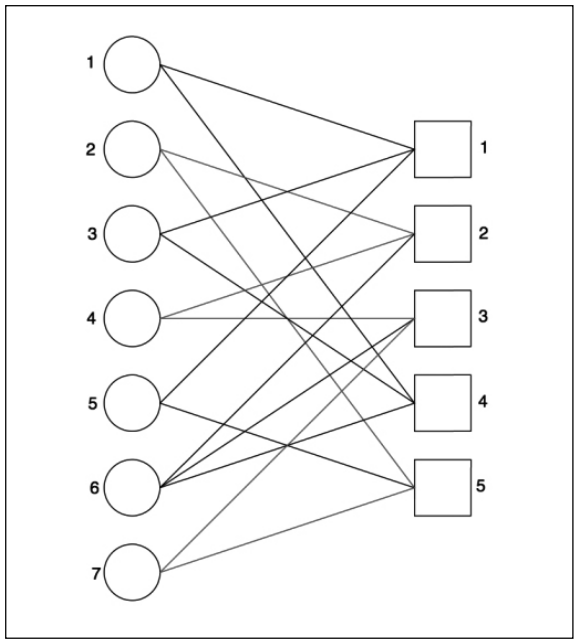

Figura 3. Conjunto de parada en el grafo de Tanner

códigos LDPC con una cintura 24 y con una pequeña variación, la cintura se incrementa hasta 28. La idea es construir un grafo que determine el conjunto de parada mínimo del código; a partir de este grafo se obtiene la matriz de incidencia la cual sería la matriz de paridad del código.

El grafo mencionado se obtiene de la siguiente manera:

1. Sea C un grafo simple, este grafo va a ser el núcleo.
2. Se realizan 2 |C| + 1 copias del núcleo, obteniendo asíi 2 |C| + 2 subgrafos.
3. Se dividen los subgrafos en 2 conjuntos: un conjunto izquierdo y un conjunto derecho, cada uno con |C| + 1 subgrafos. Se nombran los subgrafos del conjunto izquierdo como 0, 1, …, |C| y los subgrafos del conjunto derecho como 0', 1', … |C|'.
4. Se conectan los nodos usando la siguiente regla:

a. Se toma el nodo i del subgrafo j y se conecta con el nodo j del subgrafo i' para i ≠j con $1 \leq i,j \leq |C|$.

b. Se conecta el nodo i del subgrafo i al nodo i del subgrafo 0', en una manera similar, se conecta el nodo i del subgrafo i' al nodo i del subgrafo 0.

Del procedimiento anterior, se obtuvo un

grafo conexo de tamaño 2 |C| (|C| + 1). Cabe notar que el grado de cada uno de los nodos se incrementó en 1.

A modo de ejemplo, se demuestra cómo obtener un grafo regular de cintura 12, para esto, se requiere que el núcleo tenga cintura de al menos 12.

Seguidamente, se demuestra que el ciclo de longitud mínima usando la construcción anterior no es menor a 12, si la cintura del núcleo es de al menos 12. Primero que todo, es fácil ver que cada subgrafo tiene la misma cintura que el núcleo, por lo tanto, se deben analizar los ciclos formados que involucran varios subgrafos. Para hacer esto, se va a mostrar un gráfico con el menor ciclo, el cual incluye los subgrafos 0 y 0'. Sean i,j nodos adyacentes en cada subgrafo, el ciclo mostrado en la figura 4 es obtenido usando nuestro método.

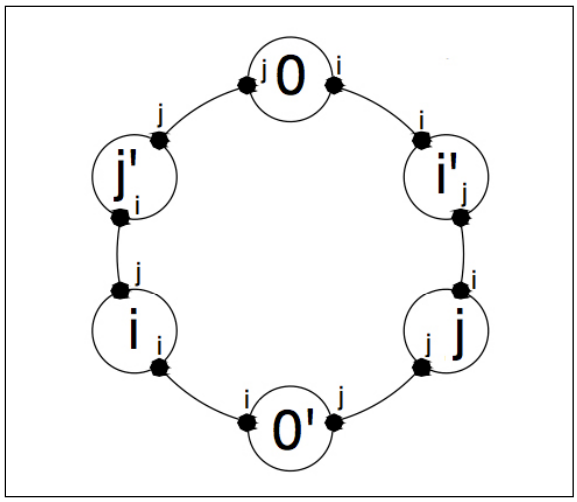

Figura 4. El ciclo más pequeño que involucra a los subgrafos 0 y 0'

Para los ciclos que no involucren los subgrafos 0 y 0', el peor de los casos genera ciclos de longitud 14. Sea i,j,k un camino en cada subgrafo, un ciclo de longitud 14 es mostrado en la figura 5.

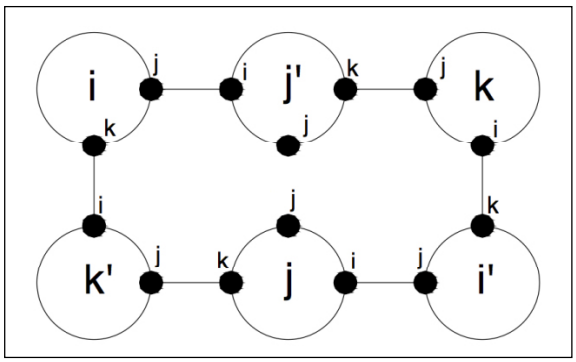

Figura 5. El ciclo más pequeño que no involucra a los subgrafos 0 y 0'

Extensión del algoritmo para obtener una cintura mayor

Para obtener una grafo que contenga una cintura 14, se requiere hacer una modificación en el método de las conexiones. Como se explicó anteriormente, se requiere que el núcleo tenga una cintura de al menos 14. La modificación consiste en hacer un renombramiento en los nombres de los nodos dentro de los subgrafos 0 y 0', de tal manera que los subgrafos codificados tengan vecinos distintos al núcleo original.

Para ver que la cintura del grafo generado es 14, hay que ver que no se puede obtener ciclos de longitud menor a 14. Como se mostró anteriormente, el ciclo más pequeño que no involucra a los subgrafos 0 y 0' tiene longitud 14. Se va a mostrar que el ciclo más pequeño que involucra a los subgrafos 0 y 0' con esta modificación tiene longitud 14. Sean i y j nodos vecinos en el núcleo original, por lo tanto no son vecinos en los subgrafos modificados, así el ciclo más pequeño que involucra a los subgrafos 0 y 0' es mostrado en la figura 6.

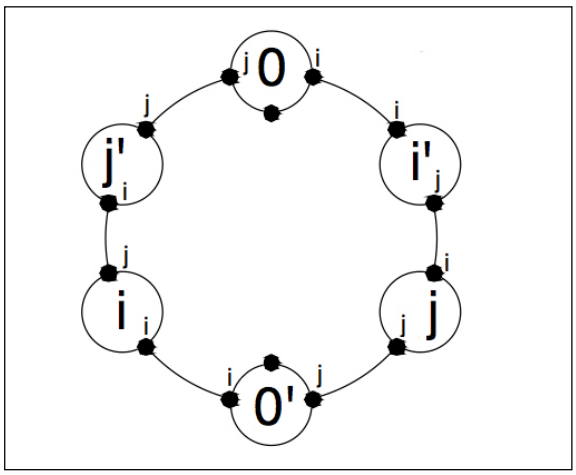

Figura 6. El ciclo más pequeño con vecinos en el núcleo original

Por otro lado, sean k y l nodos adyacentes en los subgrafos modificados, es decir, estos nodos no son vecinos en el núcleo. El ciclo más pequeño que involucra solamente 2 nodos en los subgrafos modificados tiene longitud 16, como se muestra en la figura 7.

Obteniendo el código

Como se explicó anteriormente, la matriz de paridad H del código se obtiene a partir de la matriz de incidencia. Los nodos dentro del grafo serán los nodos de chequeo y los arcos correspon-

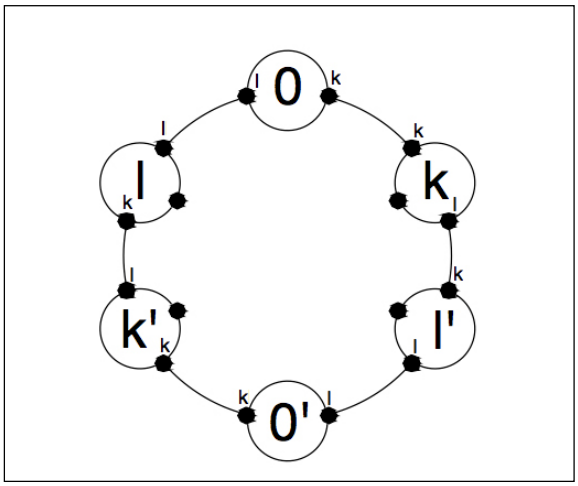

Figura 7. El ciclo más pequeño con vecinos en el núcleo modificado

den a los nodos variable. Los ciclos de longitud k generan ciclos de longitud 2k en el grafo de Tanner. Por lo anterior, el tamaño del conjunto de parada del código LDPC no va a ser menor a la cintura del grafo, consecuentemente el método garantiza que el tamaño del conjunto de parada no sea pequeño.

Si un grafo regular es escogido como el núcleo y dv es el grado de cada nodo, entonces el número de arcos dentro del grafo generado es igual a (dv + 1) |C| (|C| + 1). La tasa del código LDPC generado por este método es $R = \frac{1}{\cdot}$.

Resultados

Para nuestra simulación, escogimos un anillo de tamaño 22 como el núcleo. Este anillo es simple regular de cintura 22. Para hacer el renombramiento de los nodos el los subgrafos 0 y 0', usamos una fórmula simple: i := i*5mod(22). Para el caso del ejemplo considerado, la permutación resulta: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22] →[9, 18, 5, 14, 1, 10, 19, 6, 15, 2, 11, 20, 7, 16, 3,1 2, 21, 8, 17, 4, 13, 22].

El grafo generado tiene 2(22)(23) = 1012 nodos y 3(22)(23) = 1518 arcos, por lo tanto, la matriz de paridad H tiene tamaño 1012 x 1518, donde cada columna tiene exactamente dos 1's.

El rendimiento del código LDPC obtenido sobre un canal binario de borrado se muestra en la figura 8.

Conclusiones

Se ha presentado un nuevo algoritmo para

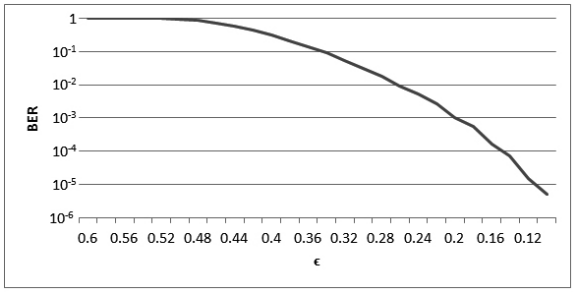

Figura 8. Rendimiento del código en un BEC (R=1/3, n=1518)

construir códigos LDPC partiendo de la generación de un grafo. Este grafo se genera como copias de un núcleo dado. Debido a que el conjunto de parada de un código LDPC está relacionado a la cintura del grafo, se obtiene un conjunto de parada de tamaño grande. La matriz de chequeo de paridad es muy dispersa, lo que resulta en una rápida convergencia del algoritmo. Las simulaciones muestran que se obtiene un código LDPC con buen rendimiento. También se debe tener en cuenta que es posible generar códigos más grandes utilizando el grafo obtenido como un nuevo núcleo.

Referencias

[1] MACKAY ,D. J. C. y NEAL, R. M. (1996). Near Shannon limit performance of low density parity check codes, Electronics Letters, 32 (18):1645–1646

[2] GALLAGER, R. G. (1962). Low-density parity-check codes, IRE Transactions on Information Theory, 8(1):21–28, Jan. 1962.

[3] RICHTER, G. (2006). Finding small stopping sets in the tanner graphs of ldpc codes, in In 4th International Symposium on Turbo Codes and Related Topics.

[4] ROSNES, E. y YTREHUS, O. (2009). "An efficient algorithm to find all small-size stopping sets of low-density paritycheck matrices," IEEE Trans. Inf. Theor., 55(9):4167–4178. [Online]. Available: http://dx.doi.org/10.1109/TIT.2009.2025573

[5] KRISHNAN, K. M. and SHANKAR, P., "On the complexity of finding stopping distance in tanner graphs," CoRR, vol. abs/cs/0512101, 2005.

[6] RICHTER, G. and HOF, A. (2006). On a construction method of irregular ldpc codes without small stopping sets. In ICC. IEEE, pp. 1119–1124.

[7] LAURI, J. and TJHAI, C.J. (2011). Coset graphs for low density parity check codes: performance on the binary erasure channel, IET Comm., 5(5):719–727.

[8] CRADDOCK, J., FLANAGAN M. F., REDMOND S. J., and FAGAN A. D., "Construction of girth 8 ldpc codes based on multidimensional finite lattices." in ISCC. IEEE, 2007, pp. 655–659.

[9] WASSINGER, N., LIBERATORi, M., and MOREIRA, J. C. (2013) Variación de la performance de decodificadores ldpc de distancia euclidiana con la base logarítmica utilizada, Revista Argentina de Ingeniería, 1(1):75–83.