Inmersión en Hipercubos (*Cubos n-dimensionales*)

Ricarda Tola

Universidad Autónoma Tomás Frías
Potosí, Bolivia
e-mail: rictopa@hotmail.com

Resumen

En la computación paralela, la arquitectura óptima depende del algoritmo diseñado para resolver un problema concreto. Una arquitectura de árbol puede ser ideal para resolver el problema X, mientras para el problema Y la malla puede ser la mejor. Por tanto, para resolver estos dos problemas, se requerirían dos computadoras paralelas con dos distintas arquitecturas. En este artículo se presenta un algoritmo, basado en la estrategia llamada de inmersión estricta, para sumergir árboles en hipercubos, lo que permite "traducir" los algoritmos diseñados para trabajar sobre la primera arquitectura, para que se puedan ejecutar sobre la segunda.

Palabras clave: Programación paralela, hipercubos, estrategia de inmersión en grafos.

1. Introducción

En las últimas décadas, hemos asistido al crecimiento continuo de las capacidades y rendimiento de los sistemas de cómputo debido a dos tipos de cambios: tecnológicos y arquitecturales. Dentro de los cambios arquitecturales, el más destacado es, sin lugar a dudas, la aparición de las computadoras paralelas.

La programación paralela involucra muchos aspectos que no se presentan en la programación convencional (secuencial). El diseño de un programa paralelo tiene que considerar, entre otras cosas, el tipo de arquitectura sobre el cual se va a ejecutar el programa, las necesidades de tiempo y espacio de la aplicación, el modelo de programación paralela adecuado para implantar la aplicación y la forma de coordinar y comunicar diferentes procesadores para que trabajen juntos en la resolución de un problema. En cuanto a las diferentes arquitecturas, las más frecuentes son la *malla* bidimensional (*grid*), el *árbol* y el *hipercubo* o *cubo* $n-dimensional$. Todas ellas son casos particulares de grafos, cuyos vértices representan los procesadores y cuyos ejes representan las conexiones físicas entre ellos, por lo que la teoría general de grafos es muy útil en su estudio.

Si se tiene un algoritmo paralelo diseñado para una cierta arquitectura, por ejemplo de malla, y se dispone de una máquina con otra arquitectura, digamos de hipercubo,

para ejecutarlo, se debe seleccionar una subred (o subgrafo) del hipercubo que constituya una malla y ejecutar el algoritmo sólo en los procesadores que componen la subred, sin necesidad de modificarlo. Este proceso se conoce como *inmersión de la malla en el hipercubo*.

Los hipercubos, o *cubos $n-dimensionales$*, son utilizados en muchos algoritmos paralelos por sus ventajas a la hora de enviar mensajes entre procesadores en el menor tiempo posible, por ello en la presente propuesta se ha tomado como la arquitectura destino para la inmersión de árboles binarios completos (CBT), que son estructuras bidimensionales y, por tanto, más simples y más comprensibles.

Las máquinas paralelas constan de varios procesadores interconectados entre sí mediante una red fija a través de la cual intercambian la información necesaria para la ejecución de los programas. Un computador paralelo puede definirse entonces como un *"conjunto de procesadores de capacidades comparables, cooperando en la ejecución de una tarea, bajo el control de un único sistema operativo, en el que se ejecuta más de un cómputo al mismo tiempo en varios procesadores"* [1].

Partiendo del hecho que un computador paralelo se puede reducir a un conjunto de elementos que cooperan para lograr un fin común, es evidente que ha de existir un medio, la red de interconexión, que facilite el intercambio de información. Una red de interconexión estática es aquella cuya topología (anillo, malla, árbol, hipercubo, etc...) queda definida y establecida durante la construcción de la máquina paralela [3].

2. Definiciones

Topología árbol

Los árboles son grafos no direccionados, acíclicos y conectados. Si se elige un nodo para distinguirlo y llamarlo raíz, se dice que el árbol está enraizado. Los **árboles enraizados** constituyen una clase muy importante de grafos para la comunicación, por su estructura jerárquica, en la que se tiene como principal al vértice raíz, mientras todos los demás son sus "descendientes". A los vértices que están a distancia 1 de la raíz se los conoce como sus **hijos** y se dice que la raíz es su **padre**. Esta relación se extiende a todos los vértices del árbol. Sólo la raíz no tiene padre y los vértices que no tienen hijos se llaman vértices o nodos hoja. Se conocen como nodos interiores a los que tienen padre y al menos un hijo. El **nivel** de un nodo es el número de nodos del camino que lleva desde éste hasta la raíz (sin incluirse a sí mismo). La **altura** de un árbol es el máximo de los niveles presentes en él o -si se prefiere- la máxima de todas las distancias entre la raíz y los nodos hoja.

En un árbol binario completo (CBT por sus siglas en inglés), los nodos interiores tienen exactamente dos hijos (hijo izquierdo e hijo derecho) y todos los nodos hoja están al mismo nivel. Un BCT de altura k se conoce como BCT_k y puede definirse recursivamente de la siguiente manera:

Definición 1. *Un BCT_k consta de Raíz, subárbol izquierdo y subárbol derecho. La Raíz es un vértice y los subárboles izquierdo y derecho son árboles binarios completos de altura $k-1$. Si $k=0$, el BCT_0 consta de un solo vértice que será la raíz.*

Según esta definición, es fácil construir el BCT_k a partir de un vértice v y de dos BCT_{k-1}, haciendo que la raíz del primer BCT_{k-1} sea el hijo izquierdo de v y la raíz del segundo BCT_{k-1} sea el hijo derecho de v.

Las propiedades que presentan los árboles binarios completos de altura k (CBT_k) son:

- Número de nodos: $2^{k+1} - 1$
- Profundidad: k
- Diámetro: $2k$
- Ancho de bisección: 1
- Número máximo de ejes por nodo: 3

En el presente campo, se utiliza la siguiente forma de nombrar los nodos de una árbol binario completo:

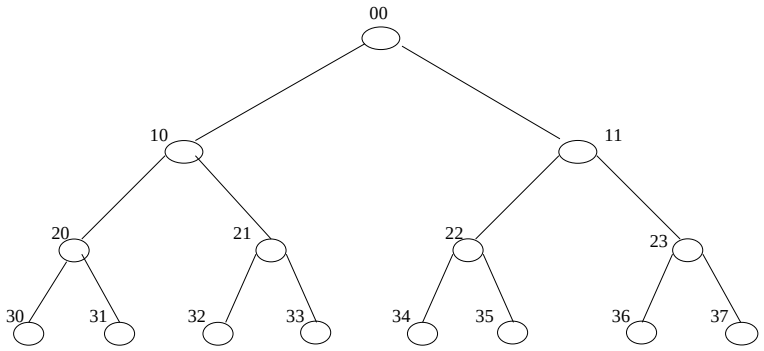

Figura 1: Árbol binario de nivel 3, CBT_3

Para la presente propuesta se requiere recorrer el CBT, visitando sistemáticamente todos los nodos del árbol. De las varias formas de recorrido de árboles, que difieren solamente en el orden en que se visitan los nodos, se ha elegido el recorrido **pre-orden**, que se define mediante una simple regla recursiva.: "visitar la raíz, después el subárbol izquierdo y a continuación el subárbol derecho".

Topología hipercubo *(cubo n–dimensional)*

Definición 2. *El hipercubo se define recursivamente en términos del producto cartesiano de grafos como sigue:*

$$Q_n = \begin{cases} K_2 & si\ n{=}1 \\ Q_{n-1}K_2 & si\ n > 0 \end{cases}$$

donde K_n es el grafo completo de orden n.

También en este caso, a partir de dos hipercubos de dimensión $k-1$, Q_{k-1}, se puede construir un hipercubo de dimensión k, Q_k, enlazando con un eje cada vértice del primer hipercubo con un vértice del segundo hipercubo.

Según la definición de grafos isomorfos

> Dos grafos G_1 y G_2 son grafos isomorfos, si existe una función biyectiva $f : V(G_1) \to V(G_2)$, tal que, si $(v_i, v_j) \in E(G_1) \Leftrightarrow (f(v_i), f(v_j)) \in E(G_2)$.

un *cubo $n-dimensional$* Q_n podría definirse también de la siguiente manera:

Definición 3. *Un hipercubo es isomorfo a un grafo G con 2^n vértices, cada uno de los cuales está etiquetado con una cadena de n-dígitos binarios, de manera tal que dos nodos son adyacentes si y solamente si sus cadenas correspondientes varían en un solo dígito.*

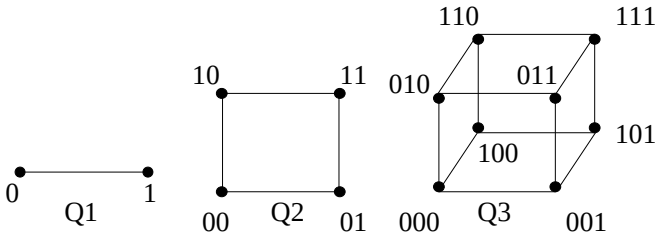

Figura 2: Hipercubos de dimensiones 1, 2 y 3

Las propiedades fundamentales de los *cubos $n-dimensionales$* son:

- El número de vértices de Q_n es 2^n.
- El número de ejes de Q_n es $n2^{n-1}$.
- Q_n es un grafo regular de grado n.
- Q_n es un grafo bipartito.
- Q_n es un grafo hamiltoniano con circuito para n$\geq$2.
- El diámetro de Q_n es n.
- Q_n es un grafo conectado.
- Ancho de bisección es 2^{k-1} [3].

Las propiedades de los hipercubos los convierten en una buena opción para constituirse en la arquitectura preferida de las máquinas paralelas de fines generales:

1. El hecho de que Q_n pueda ser definido recursivamente como el producto, $Q_n = Q_{n-1}K_2$, sugiere que un hipercubo puede utilizar la estrategia dividir-y-vencer. Ciertos algoritmos, tales como el Bitonic Sort (Ordenación) y el FFT (Transformada Rápida de Fourier) se pueden poner en ejecución eficientemente en una red hipercubo, en la cual los procesadores se comunican sólo entre pares de adyacentes.
2. El hecho de que Q_n sea homogéneo (dado cualquier par de nodos p y q, existe un automorfismo σ de Q_n para el cual $\sigma\ (p) = q$) permite que los algoritmos sean escritos de tal manera que, si se asume que un cierto nodo tiene un rol asignado, el cubo puede ser después rotado de modo que cualquier otro nodo deseado asuma ese papel.
3. El hecho de que Q_n tenga diámetro n, relativamente pequeño para el número de nodos que posee, implica que ningún mensaje entre dos procesadores arbitrarios necesite recorrer más de n puentes de comunicación (procesadores intermedios).

4. La difusión (un nodo envía la misma información a todos los demás), que es una operación crucial, puede ser implementada en n pasos, a través de la comunicación paralela conocida como "doblando recursivamente".

5. El hecho de que Q_n sea *n-conectado* sugiere que la red tenga un alto grado de *tolerancia a fallas.*

Un aspecto importante del uso eficiente de un sistema hipercubo para resolver un problema dado, es que la asignación de subtareas a los distintos procesadores supone costos de comunicación bajos. Las subtareas y sus requisitos de intercomunicación se pueden modelar por un grafo y la asignación de éstas a los procesadores son vistos como una inmersión del grafo de las tareas en el grafo de la *red hipercubo.*

3. Inmersión estricta en grafos

Definición 4. *Una inmersión (estricta) de un grafo $G = (V, E)$ en un grafo $G' = (V', E')$, es una aplicación $\Phi : G \leftarrow G'$ que consiste en dos aplicaciones:*

Φ_V*:* $V_G \rightarrow V_{G'}$*, inyectiva, (si* $u \in V_G$*, entonces* $\Phi_V(u) \in V_{G'}$ *) y*

Φ_E*:* $E_G \rightarrow E_{G'}$*, que hace corresponder a cada eje* $(u, v) \in E_G$ *un eje* $(\Phi_V(u), \Phi_V(v)) \in E_{G'}$.

Definición 5. *Un grafo G se llama cúbico, si para algún n, existe una inmersión de G en Q_n y la dimensión cúbica de G, denotada por $cd(G)$, es el menor entero positivo n para el cual G se puede sumergir en Q_n.*

Los investigadores I. Havel y J. Morávek (citados en [2]) han demostrado que un grafo G conectado es inmersible en Q_n si y sólo si es posible etiquetar todos los ejes de G con un entero $i \in$ a $\{1, 2, ..., n\}$ de tal manera que cumplan las siguientes condiciones:

i) Los ejes incidentes en un nodo tienen etiquetas diferentes.

ii) Para todo camino no cíclico en G, existe una etiqueta i que pertenece a $\{1, 2, ..., n\}$, que aparece en el camino un número impar de veces.

iii) Para todo camino cíclico de G, ninguna etiqueta i que pertenece a $\{1, 2, ..., n\}$ aparece en el camino un número impar de veces.

Sobre esta base se demuestra, por ejemplo, que el grafo $K_{1,m}$ (llamado grafo estrella) de $m+1$ nodos es cúbico y $cd(K_{1,m}) = m$, por tanto $K_{1,m}$ es sumergible en Q_m. En la figura 3 se ve un grafo $K_{1,4}$ inmerso en un Q_4.

4. Estrategia de inmersión estricta

A continuación, se presenta la estrategia propuesta para una inmersión específica de un árbol binario completo de altura n (CBT_n) en un *cubo* $(n+2) - dimensional$ Q_{n+2}.

En primer lugar se debe determinar que el árbol CBT_n es inmersible en el *cubo* $(n+2) - dimensional$ $Q_{n+2,}$, en base a las bases explicadas en el apartado anterior:

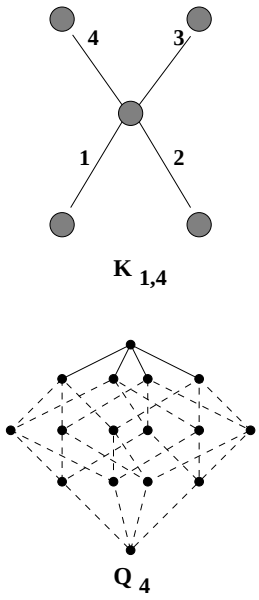

Figura 3: El grafo $K_{1,4}$ es inmersible en Q_4

1° Asignar etiquetas a los ejes de CBT_n de forma tal que cumplan las condiciones enunciadas en la sección 3, para determinar la existencia del k mínimo que indica la dimensión del *cubo* $k-$*dimensional* Q_k, en el que se va a sumergir el árbol.

En segundo lugar hay que ver cómo puede sumergirse el árbol en el hipercubo:

2° Asignar etiquetas a los ejes del *cubo* $(n+2)$ *dimensional* Q_{n+2} , y

3° Construir la función de inmersión Φ.

La figura 4 muestra que el árbol binario completo de nivel 2, CBT_2 es inmersible en el *cubo* $4-$*dimensional* Q_4, esto es CBT_2 es cúbico y $\mathrm{cd}(CBT_2) = 4$, ya que no podría sumergirse en el hipercubo de dimensión 3.

Algoritmo de inmersión estricta

Dados el grafo CBT_n y el *cubo* $(n+2) -$ *dimensional* Q_{n+2};

1° **Etiquetar los ejes del árbol** CBT_n.

Sea v la raíz de un subárbol cualquiera de CBT_n en el nivel m, $(0 \leq m \leq n)$ y sean $l(v)$ y $r(v)$ los hijos izquierdo y derecho de la raíz v. Etiquetar, los ejes que conectan v con sus hijos:
Etiq $(v,r(v)) = n - m + 2$ **y Etiq** $(v, l(v)) = n - m$.

El etiquetado del árbol CBT_n cumple con las condiciones enunciadas en la sección 3. Esta aseveración se demuestra por inducción a partir de $k = 1$.

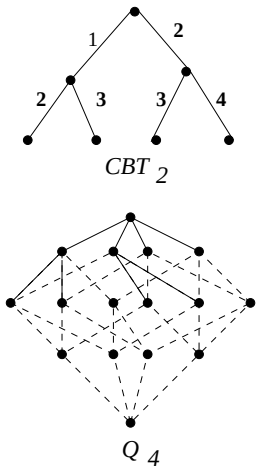

Figura 4: El árbol CBT_2 es inmersible en Q_4

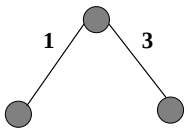

Figura 5: Árbol CBT_1 etiquetado

Prueba
Supóngase cierto para un árbol CBT_k , con $k = m-1$. Un árbol CBT_k con $k = m$, es constituido por dos subárboles CBT_{k-1}, cuyas raíces serán los hijos izquierdo ($l(v)$) y derecho ($r(v)$) de un nuevo vértice v, conectados por dos ejes que deberán ser etiquetados.

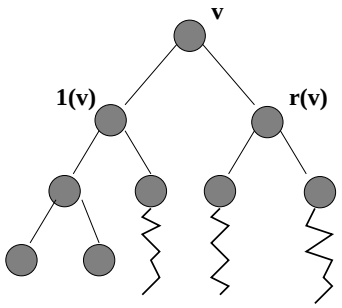

Figura 6: Árbol CBT_n enraizado en el vértice v.

Sean los ejes etiquetados como: $(v, r(v)) = n-0+2 = n+2$ y $(v, l(v)) = n-0 = n$. El subárbol cuya raíz es $r(v)$, cumple con las condiciones de la sección 3 (por hipótesis de inducción), y la etiqueta de $(v, r(v)) = n+2$ es un valor nuevo que no se repite en todo el árbol, por lo que, cualquier camino nuevo (que contenga a

este eje), cumple todavía las condiciones estipuladas.

Ahora bien, el eje $(v, l(v)) = n$ podría contradecir a las condiciones requeridas; sin embargo este valor, por el algoritmo de etiquetado, se presenta en el nivel 2 un número par de veces, ya que el árbol es binario completo, con lo que se garantiza el cumplimiento de las condiciones mencionadas.

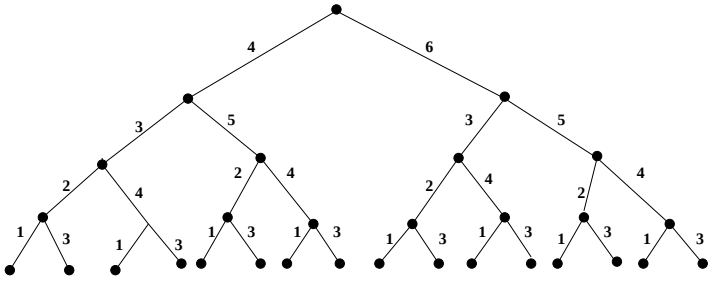

Figura 7: Árbol CBT_4 etiquetado.

2° **Etiquetar los ejes del Q_{n+2}.**

Para etiquetar los ejes de Q_{n+2}, cada nodo debe ser representado por una cadena de $(n+2)$ *bits*, de manera que el eje incidente a 2 nodos pueda ser enumerado (etiquetado) por la posición en que difieren las respectivas cadenas, lo que conduce a que los ejes incidentes a un nodo se etiqueten desde $1, 2, 3, ..., n+2$.

3° **Construir la función de inmersión**

Para construir la función de inmersión, los nodos de Q_{n+2}, son representados como una cadena de $(n+2)$ *bits*. Sea el recorrido del árbol CBT_n en $pre-orden$.

La imagen de la raíz puede ser asignada a cualquier nodo del cubo Q_{n+2}.

Para determinar la imagen del vértice v a ser visitado, se debe:

i) Identificar la imagen de su padre (que ya fue determinado), es decir, $g(v) = q_i$ que pertenece a Q_{n+2}.

ii) Identificar la etiqueta de v con su padre, $e(v)$.

iii) Identificar q_j, nodo adyacente a q_i, que difiere de los bits de este, en la posición $e(v)$ (dicho nodo es único por las propiedades de Q_{n+2}).

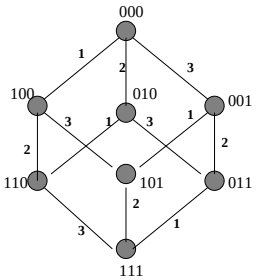

Figura 8: Etiquetado de Q_3.

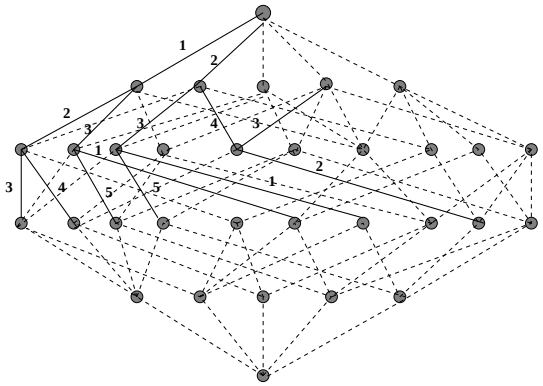

Figura 9: CBT_3 inmerso en Q_5.

iv) $g(v) = q_j$.

5. Conclusiones

1. El algoritmo de inmersión estricta desarrollado, sumerge un árbol binario completo en un hipercubo, ya que cumple las condiciones que Livinstone y Scout demostraron suficientes. Esto permite traducir los algoritmos creados para la arquitectura de árbol binario completo a algoritmos que funcionen sobre topología de hipercubo.

2. El hecho de que algunos miembros de varias clases importantes de grafos puedan verse como subgrafos de Q_n sugiere que los hipercubos pueden comportarse como esas clases ellos sin un aumento considerable de recursos, si los algoritmos diseñados para esta arquitectura se pueden adaptar fácilmente al hipercubo.

Referencias

[1] V. Kumar, A. Grama, A. Gupta, y G. Karypis. *Introduction to Parallel Computing.* Benjamin Cummings Publishing Co., Redwood City, California, 1994.

[2] M. Livingston y Q. F. Stout. Embeddings in hypercubes. *Mathematical and Computational Modelling*, 11:222–227, 1988.

[3] Ricarda Tola P. Un estudio sobre cubos n- dimensionales. Tesis de Licenciatura, Universidad Autónoma Tomás Frías, 1999.

[4] Ricarda Tola P. Inmersión en hipercubos. Tesis de maestría, Universidad Autónoma Tomás Frías, 2002.

[5] Javier Cañas R. *Apuntes de Computación Paralela.* Departamento de Informática, utfsm, 1998.

[6] Robert Sedgewick. *Algoritmos en C++.* Addison - Wesley de Santos, 1994.