Enfoques de paralelización de la reconstrucción 3D de imágenes del árbol coronario *

R. Rivas† Y. Cardinale† M. B. Ibáñez†

P. Wyndyga‡

† Départamento de Computación y Tecnología de la Información

Universidad Simón Bolívar Apartado 89000, Caracas 1080–A, Venezuela

{rivas,suarez,yudith,ibanez}@usb.ve

‡ Computer Science Department University of Central Florida

Orlando, FL 32816-2362

pwindyga@cs.ucf.edu

Resumen

Este trabajo estudia la conveniencia de realizar paralelización funcional y/o paralelización por data sobre un algoritmo combinatorio de tipo *branch and bound* cuyo objetivo consiste en reconstruir una imagen angiográfica a partir de dos proyecciones y el grafo que contiene todas las posibles soluciones.

*Este trabajo ha sido financiado por la Comunidad Económica Europea (Proyecto ITDC 139-82158 A2(MI) y por el Decanato de Investigación y Desarrollo de la USB (Proyecto S1-CAI-136)

1 Introducción

Una de las patologías más frecuentes que afectan las arterias coronarias es la reducción de su diámetro. Esta reducción es causada principalmente por la estenosis la cual consiste en la acumulación de lípidos en las paredes internas de las arterias. La oclusión total o parcial de las arterias trae como consecuencia la disminución del volumen de sangre irrigado a todos los tejidos del corazón y por lo tanto, la disminución del nivel de oxigenación del músculo cardíaco, condición que antecede a la ocurrencia de un infarto. La localización y cuantificación de esta anomalía sólo es posible mediante el análisis visual, utilizando alguna técnica de imagenología médica.

La angiografía coronaria [1] es la técnica de imagenología médica más usada para visualizar las arterias del corazón. Mediante esta técnica se obtienen imágenes bidimensionales que presentan problemas de opacidad, superposición de estructuras y deformación inherente al proceso de proyección. La correcta interpretación de las imágenes se dificulta tanto por el carácter bidimensional de la técnica como por los problemas antes descritos.

El objetivo de la investigación es presentar al médico una visión tridimensional de las arterias coronarias a partir de dos imágenes angiográficas bidimensionales, la reconstrucción se logra utilizando un conocimiento *a priori* de las estructuras 2D y 3D. Esta reconstrucción tridimensional a partir de sólo dos proyecciones, es un problema de múltiples soluciones (ver figuras 1 y 2).

En su tesis de doctorado, el Prof. Piotr Windyga [Win94] desarrolla un algoritmo secuencial para resolver este problema utilizando un conjunto de heurísticas basadas en conocimiento de imagenología. El algoritmo fué implantado en el lenguaje de programación Prolog. Dada la naturaleza combinatoria del problema, este programa consume un tiempo de cómputo que imposibilita su aplicación.

Existen dos grandes enfoques de paralelización: funcional y por data [Fos95]. La par-

[1] Técnica de visualización radiográfica que consiste en la inyección de un líquido radio-opaco en las arterias coronarias tomando radiografías desde dos focos de proyección diferentes y obteniendo pares de imágenes bidimensionales homólogas

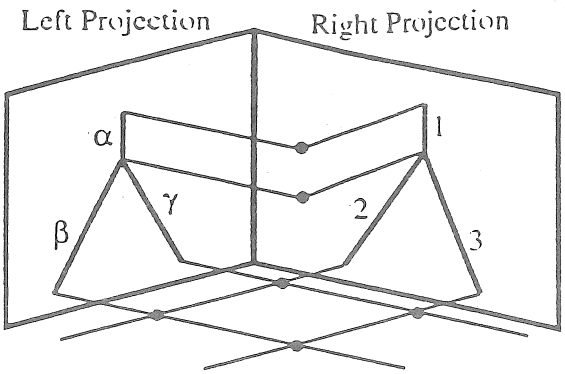

Figure 1: Espacio de Soluciones

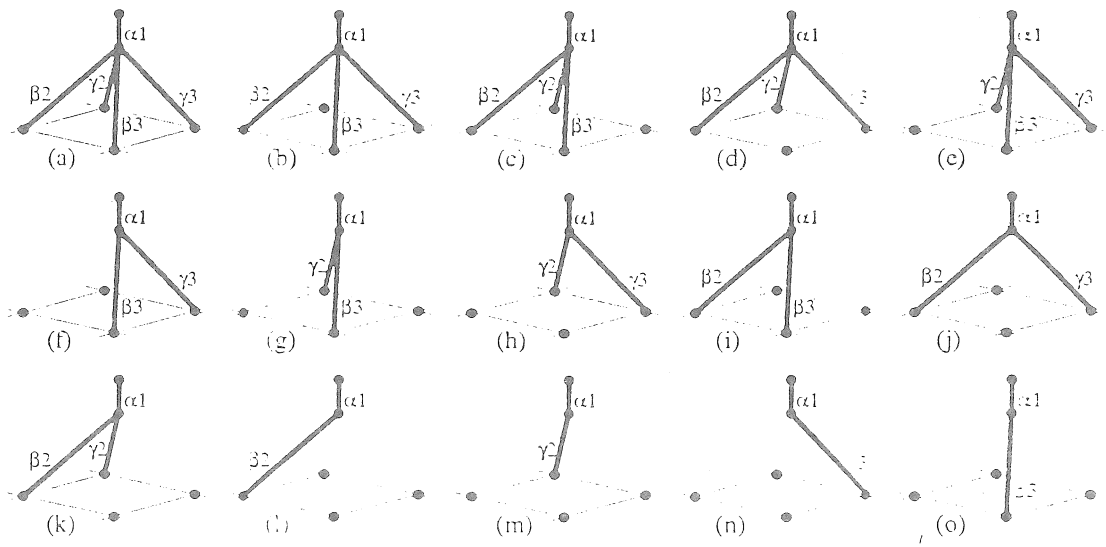

Figure 2: Árboles de Soluciones Parciales

alelización funcional consiste en realizar trabajo diferente sobre el mismo conjunto de datos. La paralelización por data consiste en realizar el mismo trabajo sobre subconjuntos de datos diferentes.

Para algoritmos de naturaleza combinatoria la paralelización por datos suele ser la más apropiada [KGGK94], pero se hace necesario cuidar la estrategia de repartición de trabajo entre los procesadores involucrados. Por otra parte la independencia existente entre las diferentes heurísticas (filtros) aplicadas a cada posible solución hace pensar en la conveniencia de realizar una paralelización funcional. Son estos dos aspectos los analizados en nuestro trabajo.

Trabajamos sobre una plataforma paralela Parsytec Xplorer de ocho (8) procesadores PowerPC con capacidad de memoria RAM de 32 Mbytes cada uno.

Este artículo está organizado como sigue. En la sección 2 se describe el algoritmo secuencial. La sección 3, presenta los resultados experimentales obtenidos al ejecutar el programa. En la sección 4, se establece la estrategia de paralelización a seguir. Los resultados de la ejecución paralela son mostrados en la seccion 5. Finalmente las conclusiones y el trabajo futuro se presentan en la sección 6.

2 Descripción general del algoritmo

El grupo GBBA [2] desarrolló una herramienta para el análisis y el estudio de imágenes angiográficas: ANIA. ANIA permite la manipulación de imágenes 2D y genera el espacio de soluciones para el proceso de reconstrucción 3D. El objeto de nuestro estudio es el proceso de reconstrucción 3D y el algoritmo que lo realiza se describe en esta sección.

El algoritmo recibe como entrada un grafo que contiene todas las representaciones tridimensionales posibles del árbol coronario (*Grafo de Soluciones Totales* : *GST*) y produce como salida un conjunto de soluciones válidas (*Árboles de Soluciones Parciales* : *ASP*). Una solución posible es válida si y solo si satisface el conjunto de filtros que se describen a continuación.

Ramificación permitida La única ramificación permitida es la de bifurcación.

Proyección Vectorial Debe verificarse que en la construcción de la combinación de arterias, ninguna de ellas se *devuelva* pues ninguna arteria puede ir en sentido contrario al del flujo sanguíneo ni tampoco atravesar el músculo cardíaco.

Coplanaridad aceptable Las nuevas arterias añadidas no pueden formar un plano con las que ya pertenezcan a la solución pues el corazón tiene forma ovoidal.

Superposición aceptable Las nuevas arterias agregadas a la solución deben tener una *superposición aceptable*, se estima que la superposición no debe superar los dieciocho (18) milímetros.

[2] Grupo de Bioingeniería y Biofísica Aplicada de la Universidad Simón Bolívar

En lo que sigue utilizaremos la siguiente nomenclatura. Un $GST =< \mathcal{P}, \mathcal{A} >$ donde $\mathcal{P} = \{p_1, \ldots p_m\}$ (p_i son puntos del GST) y $\mathcal{A} = \{a_1, \ldots a_n\}$ (a_i son aristas del GST). Un ASP $=< \mathcal{P}', \mathcal{A}' >$ donde $p'_i \in \mathcal{P}'$ y p'_i es una copia de $p_i \in \mathcal{P}$, de la misma manera $a'_i \in \mathcal{A}'$ y a'_i es una copia de $a_i \in \mathcal{A}$. Una arista $a =< p_k, p_l >$ es incidente a los puntos p_k and p_l que se denominan sus *puntos extremos*. Los puntos en la frontera del ASP se denominan *Puntos Activos* : $\mathcal{PA}'$.

Algoritmo

1. El primer ASP se construye a partir del *Grafo de Soluciones Totales* comenzando con un punto inicial y sin aristas, i.e. $\mathcal{P}' = \mathcal{PA}' = \{p'\}$, $\mathcal{A}' = \{\}$.

2. Mientras haya *Puntos Activos* ($\mathcal{PA}' \neq \{\}$) haga

   (a) $\forall\ a'_i \in \mathcal{PA}'$ construya $\mathcal{N}_i = \{a'_{i1}, \ldots, a'_{ir}\}$ donde $a_{i1}, \ldots, a_{ir}$ son incidentes a a_i y ninguno de ellos tiene un punto extremo en $\mathcal{P}'$.

   (b) $\forall\ j_1, \ldots, j_n \in [1, 2]$: construya nuevos ASP s usando las siguientes reglas:

      i. $\mathcal{A}'_{nuevo} = \mathcal{A}' \bigcup \mathcal{N}_1^{j_1} \times \ldots \times \mathcal{N}_n^{j_n}$

      ii. $\mathcal{PA}'_{nuevo}$ contiene a los nuevos puntos de la frontera del ASP_{nuevo}

      iii. $\mathcal{P}'_{nuevo}$ es el conjunto de los puntos extremos de $\mathcal{A}'_{nuevo}$

      Descarte el ASP actual.

   (c) Valide las nuevas aristas añadidas usando los filtros de Proyección Vectorial, Coplanaridad aceptable y Superposición aceptable.

      Si algún filtro falla, el ASP_{nuevo} correspondiente es descartado.

La generación combinatoria de soluciones se produce en la etapa 2b del algoritmo y puede llevarse a cabo en paralelo. Nótese que para cada *Punto Activo* se consideran a lo sumo dos aristas, es ahí donde se trabaja la *Ramificación permitida*. En la etapa 2c del algoritmo se aplican los tres filtros restantes, debido a que son independientes pueden ser aplicados en paralelo.

3 Comportamiento del programa secuencial

Debido a la imposibilidad de paralelizar el programa original escrito en Prolog sobre nuestra plataforma de trabajo, se reescribió el algoritmo en lenguaje C. En esta sección se muestran los resultados obtenidos en nuestra máquina paralela.

Los datos de entrada a nuestro programa corresponden a la sección superior de arterias de un corazón real. La herramienta ANIA nos dá como entrada dos imágenes bidimensionales que tienen 150 y 160 puntos respectivamente, y el *Grafo de Soluciones Totales* con el que trabajamos tiene 120 segmentos.

El experimento se realiza dando como entrada al programa secuencial 40, 80, 100 y 120 segmentos del *Grafo de Soluciones Totales* Los resultados obtenidos se muestran a continuación.

Número de segmentos	Número de soluciones	Tiempo en seg	Complanaridad (promedio seg)	Proy. Vectorial (promedio seg)	Superposición (promedio seg)
40	849	0.6840	13.82561e-05	2.125588e-05	8.945256e-05
80	28113	43.9537	26.47590e-05	2.488055e-05	8.949674e-05
100	120476	229.3084	42.06210e-05	2.402490e-05	8.961020e-05
120	293231	594.0548	53.96585e-05	1.796452e-05	12.44755e-05

Tabla 1: Desempeño del programa secuencial

En la tabla 1 puede apreciarse cómo a medida que aumenta el número de segmentos, aumenta de manera exponencial el tiempo de ejecución del programa y aumenta de manera lineal el tiempo consumido por el filtro de Coplanaridad. Los otros dos filtros mantienen su promedio de tiempo consumido y la suma de ambos es inferior al requerido por Coplanaridad.

Se aprecia que el número de soluciones válidas es muy grande, se hace necesario realizar nuevas investigaciones que permitan incluir nuevos filtros al programa que permitan reducir el número de estas soluciones.

4 . Estrategia de paralelización

El algoritmo secuencial (ver sección 2) construye, a partir de un grafo inicial (*Grafo de Soluciones Totales*), un conjunto de árboles (*ASP*) sobre los que se aplican las heurísticas que determinan si los *ASP* corresponden a soluciones válidas.

El proceso de construcción de los *ASP* es combinatorio y es el responsable principal del mal desempeño de las implantaciones secuenciales del algoritmo. En un escenario ideal donde conociésemos el número de *ASP* a construir y tuviésemos forma de sistematizar el proceso de generación de estos árboles, sería factible resolver adecuadamente el aspecto crítico en la paralelización de algoritmos de naturaleza combinatoria: la distribución del espacio de búsqueda entre los procesadores.

Al era bien, aún cuando el *Grafo de Soluciones Totales* es conocido de antemano, el problema de calcular el número de combinaciones posibles a partir del grafo resulta de complejidad semejante al problema que nos ocupa. Estamos pues ante una situación en la que debemos dividir el espacio de búsqueda a ciegas. La consecuencia natural de esto es que se producirá un desbalance de carga, dependiendo de cuán grande sea, tendrán que tomarse medidas para corregirlo.

Nuestro algoritmo inicia su recorrido en la parte del *GST* que corresponde a las arterias principales del corazón y éstas se bifurcan de manera de irrigar equitativamente al músculo cardíaco. Esto nos hace pensar que dividiendo el trabajo a realizar desde el inicio del grafo es posible tener un balance de carga equilibrado. En este trabajo realizaremos esta estrategia de división del espacio de búsqueda estática y mediremos el desbalance de carga que esta división origina.

Es también posible definir una estrategia de paralelización funcional. A cada posible solución es necesario aplicar tres filtros: *Proyección Vectorial*, *Superposición Aceptable* y *Coplanaridad Aceptable*. En la sección 3 se muestra que este último siempre consume más tiempo que la suma de *Proyección Vectorial* y *Superposición Aceptable*.

Estimamos que puede resultar beneficioso agrupar los procesadores en parejas que trabajen cada una de ellas sobre las mismas unidades de trabajo. Los procesadores pares aplicarán

los filtros *Proyección Vectorial* y *Superposición Aceptable* en tanto que los impares aplicarán *Coplanaridad Aceptable*. Estas parejas de procesadores se comunicarán tan solo en el evento en el que algumo de los filtros falle, de manera de descartar como válida la posible solución.

Los procesadores pares serán los encargados de generar y solicitar nuevas unidades de trabajo.

5 Comportamiento del programa paralelo

La tabla 2 muestra los tiempos empleados por el programa secuencial y el programa paralelo por datos para 40, 80, 100 y 120 segmentos. Allí puede apreciarse que efectivamente hay un mejor desempeño del programa paralelo que del secuencial, la mejora llega a alcanzar un 68.95%.

Número segmentos	Tiempo 1 Procesador	Tiempo 2 Procesadores	Tiempo 4 Procesadores
40	0.6846	0.3427	0.1886
80	43.9537	24.9355	13.5908
100	229.3054	132.6911	73.8781
120	594.0543	338.0884	184.4802

Tabla 2: Comparación de desempeño del programa secuencial con el programa paralelo por datos con 2 y 4 procesadores

Las tablas 3 y 4 muestran el desempeño del programa paralelo por datos para 2 y 4 procesadores respectivamente con los mismos datos de prueba. El experimento muestra cómo a mayor número de segmentos considerados se observa un peor balance de carga, esto se mantiene tanto para 2 como para 4 procesadores. El caso extremo lo tenemos en el experimento con 4 procesadores y 100 segmentos donde alcanza un desbalance máximo del 42.31%. Esto nos indica que en este problema se puede obtener un mejor desempeño si se cuida el aspecto de balance de carga.

Número de segmentos	Procesador	Número de soluciones	Tiempo en seg
40	P0	447	0.341139
40	P1	462	0.342733
80	P0	12410	18.715004
80	P1	15763	24.935527
100	P0	52350	94.805848
100	P1	68125	132.691112
120	P0	129533	251.971087
120	P1	163413	338.088418

Tabla 3: Desempeño del programa paralelo por datos para 2 procesadores

El tiempo de comunicación por datos utilizando la prueba patrón de *ping-pong* es de 0.000412 segundos. Esto afecta el desempeño del programa paralelo funcional.

La tabla 5 muestra los resultados de la paralelización funcional utilizando dos procesadores y 40, 80, 100 y 120 segmentos y su comparación con los resultados obtenidos por el programa secuencial. Como puede observarse el desempeño del programa secuencial es mejor que el del programa paralelo funcional. Esto se debe a que el tiempo de comunicación de datos es mayor que el que usan los filtros sobre una posible solución (ver tiempos de cómputo promedios utilizados por el filtro que consume la mayor cantidad de tiempo: Coplanaridad en la tabla 1).

6 Conclusiones y trabajo futuro

El programa secuencial actual permite hacer experimentos que no podían ser realizados con el programa escrito en Prolog. La primera conclusión que se extrae es que es necesario incluir nuevos filtros para poder obtener un número menor de soluciones válidas (Ver Tabla 1 de la sección 3). El grupo de P. Windyga está realizando investigación en este sentido.

Las pruebas realizadas sobre la paralelización funcional muestran que no es conveniente

realizarla para los filtros actuales. Sin embargo, conviene tomarla en cuenta una vez que se incluyan nuevos filtros al sistema. Los procesadores deben realizar cómputo que sobrepase el tiempo de comunicación de mensajes para que la paralelización funcional ayude a mejorar el desempeño del programa.

Por último, la paralelización por data es la más idónea para este problema. Sin embargo, se ve la necesidad de corregir el desbalance de carga existente. Este es el próximo paso en nuestra investigación.

Referencias

[Fos95] I. Foster. *Designing and building parallel programs. Concepts and tools for Parallel Software Engineering.* Addison Wesley, 1995.

[KGGK94] V Kumar, A Grama, A Gupta, and G Karypis. *Introduction to parallel computing. Design and analysis of algorithms.* The Benjamin/Cummings, 1994.

[PM95] G. Passariello and F. Mora. *Imagenología Médica.* Equinoccio, 1995.

[WBCG95] P. Windyga, G. Bevilacqua, J.L. Coatrieux, and M. Garreau. Estimation of search-space in 3D coronary artery reconstruction using angiographic biplane images. In *XVII IEEE-EMBC Conference*, pages 389–390, Montreal, Canada, 1995. IEEE.

[Win94] P. Windyga. *Evaluation et modelisation de connaissances pour la reconstruction tridimensional du reseau vasculaire cardiaque en angiographie biplan.* PhD thesis, Presentada a la Universidad de Rennes I. Rennes, Francia, 1994.

[WLB+] P. Windyga, I. López, G. Bevilacqua, M. Garreau, and J.L. Coatrieux. Utility of 2D properties in the reconstruction of coronary arteries from biplane angiographic images. In *XVII IEEE-EMBC Conference*.

Número de segmentos	Procesador	Número de soluciones	Tiempo en seg
40	P0	188	0.16376
40	P1	202	0.15254
40	P2	245	0.18860
40	P3	214	0.17888
80	P0	8630	13.5908
80	P1	6586	9.58667
80	P2	5821	9.11740
80	P3	7073	11.3720
100	P0	30857	73.8781
100	P1	28821	52.0511
100	P2	23538	42.6252
100	P3	30268	58.8901
120	P0	88823	184.4802
120	P1	67242	130.6144
120	P2	62591	120.7492
120	P3	74635	153.7724

Tabla 4: Desempeño del programa paralelo por datos para 4 procesadores

Número de segmentos	Tiempo secuencial	Tiempo paralelo
40	0.6840	1.647448
80	43.95387	78.330871
100	229.3084	357.512485
120	594.0548	851.258905

Tabla 5: Comparación de desempeño de los programas secuencial y funcional paralelo