



Universidad de Concepción
Dirección de Postgrado
Facultad de Ingeniería - Programa de Magíster en Ciencias de la Computación

MÉTODO EFICIENTE DE CLUSTERING DE FIBRAS CEREBRALES BASADO EN DISTRIBUCIÓN DE PUNTOS

Propuesta de tesis para optar al grado de
MAGÍSTER EN CIENCIAS DE LA COMPUTACIÓN

POR

Andrea Vázquez Varela
CONCEPCIÓN, CHILE

Abril 2019

Profesores guías: Pamela Guevara Alvez, Cecilia Hernández Rivas
Departamento de Ingeniería Informática y Ciencias de la Computación
Facultad de Ingeniería
Universidad de Concepción

AGRADECIMIENTOS

A Jonito, Lana, Carmucha y Pirri, por ser tan simpáticos. A Champa, por hacerme compañía durante el último año de Magíster.

A mis tutoras Pamela y Cecilia, por guiarme, aconsejarme y ayudarme durante la tesis.

A mis padres, mi hermana, mis abuelos y Luisa, por apoyarme, aconsejarme y preocuparse por mí.

En especial a Narciso por su ayuda, cariño y comprensión, y también por motivarme y animarme en los momentos más críticos de la tesis.



Resumen

Los avances tecnológicos en la imagenología del cerebro humano han ayudado mucho en los campos de investigación de neurociencia y medicina proporcionando numerosos métodos que van desde el diagnóstico de ciertas enfermedades hasta nuevas herramientas de apoyo en neurocirugía.

Mediante la Resonancia Magnética de Difusión (dMRI) es posible obtener los tractos de fibras de la materia blanca (WM). Con la tractografía se representan las principales trayectorias de las fibras cerebrales en el espacio 3D, obteniendo la estructura de la materia blanca para diferentes poblaciones de sujetos.

La principal motivación para el trabajo propuesto en esta tesis, es la necesidad de disponer de un método de clustering de fibras de la materia blanca del cerebro. El clustering se basa en agrupar fibras, con similar forma, trayectoria y posición en fascículos, partiendo de las fibras en 3D de una tractografía. El método implementado consiste en descubrir fascículos de fibras cortas y largas en una tractografía utilizando técnicas de clustering. El algoritmo consta de cuatro etapas: construcción de los clústeres de puntos, generación de clústeres de fibras preliminares, reasignación de los clústeres más pequeños de fibras preliminares a clústeres más grandes y fusión de clústeres de fibras candidatos.

Se realizaron pruebas con sujetos de la base de datos ARCHI y se compararon los resultados con el método QuickBundles. Además, se llevó a cabo un análisis cualitativo y cuantitativo del método y mediciones del tiempo de ejecución. Para una tractografía de 2.7 millones de fibras se ejecuta aproximadamente en 8 minutos.

Como conclusión, se ha creado un nuevo algoritmo que descubre clústeres de buena calidad de fibras cortas y largas, sobre tractografías de gran tamaño. Además, mejora muchos aspectos de los algoritmos presentes en el estado del arte, tales como: la calidad de los clústeres, tiempo de ejecución e independencia de información anatómica y de software privativo.

Índice general

Resumen	III
Índice de cuadros	VII
Índice de figuras	VIII
Capítulo 1. INTRODUCCIÓN	1
1.1. INTRODUCCIÓN GENERAL	1
1.2. HIPÓTESIS	2
1.3. OBJETIVOS	3
1.3.1. OBJETIVO GENERAL	3
1.3.2. OBJETIVOS ESPECÍFICOS	3
1.4. ESTRUCTURA DE LA TESIS	3
Capítulo 2. ESTADO DEL ARTE	4
2.1. INTRODUCCIÓN	4
2.2. CONCEPTOS TEÓRICOS	4
2.2.1. Imágenes de Resonancia Magnética de Difusión	4
2.2.2. Tractografía	5
2.2.3. Clustering y segmentación de fibras	6
2.3. TRABAJO RELACIONADO	7
2.3.1. Clustering exploratorio	8
2.4. CLUSTERING DE FIBRAS	9
2.4.1. Clustering sobre conjuntos de datos fibras masivos	10
2.4.2. QuickBundles	11
2.4.3. Clustering basado en K-means sobre puntos de las fibras	16
2.5. VISUALIZACIÓN E INTERACCIÓN	19
2.5.1. Visualización con OpenGL moderno	19

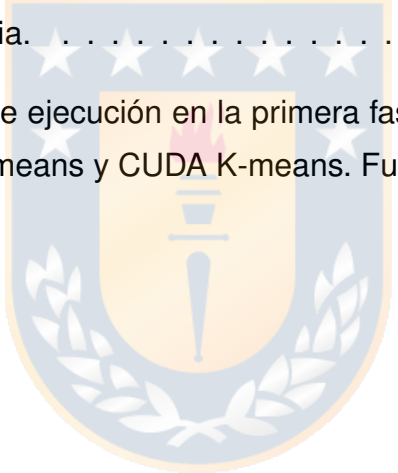
2.5.2. Interacción y visualización de grandes volúmenes de datos	20
2.6. DISCUSIÓN	23
Capítulo 3. MATERIALES Y MÉTODOS	27
3.1. INTRODUCCIÓN	27
3.2. BASE DE DATOS	27
3.3. MÉTODO DE SEGMENTACIÓN DE FIBRAS	27
3.3.1. Atlas multisujeto	29
3.3.2. Segmentación de WM basada en atlas multisujeto	30
3.3.3. Nuevo algoritmo optimizado	31
3.4. MÉTODO DE CLUSTERING DE FIBRAS	35
3.4.1. Mini Batch K-means en clustering de puntos de fibras	35
3.4.2. Método Elbow	36
3.4.3. Clique y clique maximal	38
3.4.4. Cálculo de los centroides	38
3.4.5. Nuevo método: CENTELLA	40
Capítulo 4. RESULTADOS	48
4.1. INTRODUCCIÓN	48
4.2. RESULTADOS SEGMENTACIÓN DE FIBRAS	48
4.3. RESULTADOS PARA EL CLUSTERING DE FIBRAS	50
4.3.1. Configuración de parámetros para el análisis cuantitativo	50
4.3.2. Configuración de parámetros para el análisis cuantitativo de Quick- Bundles	58
4.3.3. Comparativa con QuickBundles mediante análisis cualitativo	61
4.3.4. Comparativa en tiempo de ejecución con QuickBundles	68
Capítulo 5. CONCLUSIONES	71
5.1. CONCLUSIONES GENERALES	71
5.2. TRABAJO FUTURO	72
5.2.1. Tareas pendientes para completar el método de clustering	72

5.2.2. Inclusión del algoritmo de Clustering en un visualizador	73
5.3. PUBLICACIONES	74
Bibliografía	77



Índice de cuadros

2.1.	Comparativa del tiempo de compresión en función del MDL. Se mide el tiempo con diferentes valores MDL, desde una linealización no restringida hasta una linealización restringida hasta 5mm y con diferentes valores MET (Maximum Error Threshold), que es un umbral máximo de distancia para descartar puntos en el proceso de linealización. Fuente: Rheault et al. (2017). . .	23
4.1.	Comparativa de tiempos con QB. Se muestran los resultados del tiempo de ejecución de nuestro algoritmo y QB con distancias 6 y 10mm en segundos. Se utilizaron sujetos de la base de datos ARCHI de hasta 2,7 millones de fibras. Fuente: elaboración propia.	68
5.1.	Tiempo de ejecución en la primera fase de los algoritmos Mini-Batch K-means y CUDA K-means. Fuente: elaboración propia.	73



Índice de figuras

2.1.	Imagen de Resonancia Magnética Anatómica (contraste T1). Las zonas más oscuras representan una mayor concentración de agua y las más claras representan el tejido. Es una imagen de diferentes contrastes entre la materia blanca, la materia gris y el fluido cerebroespinal. Fuente: Haacke et al. (1999).	5
2.2.	Representación 3D de las fibras de una tractografía (cerebro completo). Esta tractografía se creó a partir de un modelo llamado deconvolución esférica limitada utilizando datos dMRI de una MRI de alta sensibilidad (el escáner Connectome) en el centro <i>Martinos Center for Brain Imaging</i> , Hospital General de Massachusetts, en Charlestown, Boston, Massachusetts. Fuente: Harvard (2018).	6
2.3.	Esquema general del método de segmentación de fibras. Paso 1 (STEP 1): Descomposición jerárquica: El conjunto de fibras se segmenta en cuatro subconjuntos principales. Paso 2 (STEP 2): Segmentación basada en longitud: Se separan las fibras de cada subconjunto en grupos que contienen fibras de tamaño similar. Paso 3 (STEP 3): Clustering basado en vóxeles: Se subdividen las fibras de cada grupo anterior a través de la parcelación de la materia blanca basada en la conectividad de las fibras. Paso 4 (STEP 4): Clustering basado en extremidades: Las fibras de los clústeres se dividen en fascículos utilizando las extremidades de las fibras. Paso 5 (STEP 5): Fusión de fascículos: Los fascículos que tienen una geometría similar se fusionan. Fuente: Guevara et al. (2011a).	12

2.4.	Fascículos resultantes para el hemisferio izquierdo de un adulto. Cada fascículo tiene un color aleatorio asignado. A: Vista externa del conjunto completo de resultados. Cada fascículo está separado en 10 grupos de diferente longitud (para facilitar la visualización). B: Vista exterior e interior de un grupo de fibras cortas. C: Vistas exterior e interior del grupo de fibras largas. Fuente: Guevara et al. (2011a).	13
2.5.	Complejidad temporal del algoritmo QB. Eje x: Número de streamlines. Eje y: segundos que tarda QB en procesar los streamlines según los diferentes umbrales de distancia. Fuente: Garyfallidis et al. (2012).	15
2.6.	Variación del número de clústeres en función del umbral de distancia mínima. En la imagen de la derecha se pueden observar los clústeres originales (verde, rojo y azul) con todas las fibras. Las imágenes del centro y derecha son el resultado de aplicar umbrales de 8 y 1 respectivamente. Cuanto menor es el umbral de distancia mínima, menos fibras acepta en los clústeres. Fuente: Garyfallidis et al. (2012).	16
2.7.	Pasos para realizar clustering: I. Se aplica el algoritmo Minibatch K-means en paralelo (multiprocesador) sobre los puntos seleccionados, obteniendo clústeres de puntos. II. Fusión de etiquetas: Los streamlines se identifican con una lista de etiquetas de los clúster a los que pertenece cada uno de sus puntos, formando así grupos de streamlines. III. Fusión de centros de clúster: Se fusionan los clústeres de los grupos de streamline que comparten el clúster del punto medio, a partir de sus centroides, después se añaden los streamlines cuya distancia no supere el umbral máximo establecido. Fuente: Sanchez et al. (2018).	17

2.8.	Centroides de los clústeres de los streamlines. En la imagen a) se muestran los streamlines correspondientes a centroides de los clústeres después de aplicar el paso 2 del algoritmo. En la imagen b) se muestran los streamlines correspondientes a los centroides después de aplicar el paso 3. Fuente: Sanchez et al. (2018).	19
2.9.	Comparativa entre colores planos y el algoritmo de iluminación de Phong para un atlas de fascículos de la materia blanca (Bonometti et al. (2015)) y un conjunto de datos de clustering. Fuente: Guevara et al. (2011a).	20
2.10.	Comparativa entre Direct Rendering y Primitive Restart. En la gráfica se muestran los tiempos de frame para las diferentes plataformas y conjuntos de datos de tractografías. Fuente: Bonometti et al. (2015).	20
3.1.	Atlas de fascículos de SWM y un sujeto segmentado con el nuevo algoritmo (vistas coronal, axial y sagital). A. Atlas de fascículos de SWM (LNAO-SWM79), compuesto de 50 fascículos por hemisferio. B. Sujeto segmentado de 955K fibras. Fuente: elaboración propia.	29
3.2.	Descarte en 4 etapas del algoritmo de segmentación. Fase 1: descarte por el punto central. Fase 2: descarte utilizando los puntos de los extremos. Fase 3: descarte utilizando 4 puntos intermedios. Fase 4: clasificación de fibras mediante la métrica de la distancia Euclidiana normalizada. Fuente: Labra et al. (2017). . .	32
3.3.	Método Elbow. El número óptimo de clústeres viene dado por el cambio en la inercia, es decir, el punto que muestra el codo del brazo (k=3). Fuente: https://pythonprogramminglanguage.com/kmeans-elbow-method/	37

3.4.	Clique maximal. El grafo G tiene dos cliques maximales, C_1 formado por los vértices $\{1, 3, 4, 5\}$ y C_2 compuesto por los vértices $\{1, 2, 6\}$. Fuente: elaboración propia.	39
3.5.	CENTELLA. PASO 1: Construcción de los clústeres de puntos. Mini-Batch K-means se aplica en paralelo en los puntos marcados, generando clústeres que posteriormente se marcan con la misma etiqueta. PASO 2: Generación de clústeres de fibras preliminares. Todas las fibras que comparten la misma etiqueta formarán el mismo clúster. PASO 3: Reasignación de clústeres pequeños a clústeres más grandes con fibras preliminares. Los clústeres se separan en conjuntos grandes y pequeños (3.1.). Se calculan los centroides (3.2.). Finalmente ocurren 3 casos: se reasignan, no se reasignan o se eliminan (3.3.). PASO 4: Fusión de clústeres de fibras candidatos. El objetivo es reducir el número de clústeres mediante cliques maximales. Fuente: elaboración propia.	47
4.1.	Tiempo de ejecución para ambos algoritmos en función del tamaño del conjunto de datos de fibras del sujeto. Fuente: elaboración propia.	49
4.2.	Uso de memoria para ambos algoritmos en función del tamaño del conjunto de datos de fibras del sujeto. Fuente: elaboración propia.	50
4.3.	Método Elbow mostrando el k óptimo. En el eje x se muestra el número de clústeres, en el eje y se muestra la inercia. Los valores óptimos de k se sitúan en el “codo” de la línea. Fuente: elaboración propia.	51

4.4.	Histograma de distancia máxima intraclúster con 100 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función de la distancia máxima intraclúster utilizando $Kp_c = 100$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 100, 150, 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.	52
4.5.	Histograma de distancia máxima intraclúster con 150 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función de la distancia máxima intraclúster utilizando $Kp_c = 150$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 150, 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.	53
4.6.	Histograma de distancia máxima intraclúster con clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función de la distancia máxima intraclúster utilizando $Kp_c = 200$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.	53
4.7.	Histograma de tamaño de clústeres con 100 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función del tamaño de los clústeres utilizando $Kp_c = 100$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 100, 150, 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.	54
4.8.	Histograma de tamaño de clústeres con 150 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función del tamaño de los clústeres utilizando $Kp_c = 150$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 150, 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.	55

4.9.	Histograma de tamaño de clústeres con 200 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función del tamaño de los clústeres utilizando $Kp_c = 200$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.	55
4.10.	Histograma de distancia máxima intraclúster con $thr_s = 4mm$ y variando el valor de thr_j en 6mm, 8mm, 10 mm y 12mm. Fuente: elaboración propia.	58
4.11.	Histograma de distancia máxima intraclúster con $thr_s = 6mm$ y variando el valor de thr_j en 6mm, 8mm, 10 mm y 12mm. Fuente: elaboración propia.	58
4.12.	Histograma de distancia máxima intraclúster con $thr_s = 8mm$ y variando el valor de thr_j en 6mm, 8mm, 10 mm y 12mm. Fuente: elaboración propia.	59
4.13.	Histograma de distancia máxima intraclúster para el método QB. Se utiliza como valores de MDF 6, 8, 10 y 12mm respectivamente. Fuente: elaboración propia.	60
4.14.	Histograma de tamaños de clúster para el método QB. Se utiliza como valores de MDF 6, 8, 10 y 12mm respectivamente. Fuente: elaboración propia.	60
4.15.	Comparación de clústeres más delgados. En la comparación se muestra nuestro algoritmo con $Kp_c = 200$, $Kp_o = 300$, $thr_s = 6mm$, $thr_j = 6mm$, junto con QB con $MDF = 6mm$ y $MDF = 10mm$. Fuente: elaboración propia.	62
4.16.	Comparación de clústeres más gruesos. En la comparación se muestra nuestro algoritmo con $Kp_c = 200$, $Kp_o = 300$, $thr_s = 6mm$, $thr_j = 6mm$, junto con QB con $MDF = 6mm$ y $MDF = 10mm$. Se comparan los tres tipos de vistas del cerebro, axial, coronal y sagital. Fuente: elaboración propia.	64

4.17.	Comparación de clústeres de fibras cortas. Los clústeres contienen los 200 clústeres más gruesos con fibras de longitud entre 30mm y 60mm. En la comparación se muestra nuestro algoritmo con $Kp_c = 200$, $Kp_o = 300$, $thr_s = 6mm$, $thr_j = 6mm$, junto con QB con $MDF = 6mm$ y $MDF = 10mm$. Se comparan los tres tipos de vistas del cerebro, axial, coronal y sagital. Fuente: elaboración propia.	65
4.18.	Comparación de clústeres de fibras largas. Los clústeres contienen los 50 clústeres más gruesos con fibras de longitud superior a 60mm. En la comparación se muestra nuestro algoritmo con $Kp_c = 200$, $Kp_o = 300$, $thr_s = 6mm$, $thr_j = 6mm$, junto con QB con $MDF = 6mm$ y $MDF = 10mm$. Se comparan los tres tipos de vistas del cerebro, axial, coronal y sagital. Fuente: elaboración propia.	66
4.19.	Imágenes de los clústeres con mayor distancia intraclúster. En nuestro algoritmo se muestran con distancia intraclúster >40mm, en QB con $MDF = 6mm$ se muestran los clústeres con distancia >50mm y en QB con $MDF = 10mm$ los clústeres con distancia >70mm. Fuente: elaboración propia.	67
4.20.	Gráfica comparativa de tiempos con QB. Se muestra la tendencia en tiempo de ejecución de los algoritmos en función del número de fibras de los sujetos, que tienen hasta 2.7 millones de fibras. Fuente: elaboración propia.	69
4.21.	Desglose de tiempos de ejecución para cada una de las etapas del algoritmo de clustering: K-means, Map, Reasignación y unión de grupos. Las ejecuciones se realizan en sujetos de la base de datos ARCHI de hasta 2,7 millones de fibras. Fuente: elaboración propia.	70

Capítulo 1

INTRODUCCIÓN

1.1. INTRODUCCIÓN GENERAL

Los avances tecnológicos en la adquisición, tratamiento, visualización y análisis de imágenes del cerebro humano han ayudado mucho en los campos de investigación de neurociencia y medicina proporcionando numerosos métodos que van desde el estudio y apoyo al diagnóstico de ciertas enfermedades hasta nuevas herramientas de apoyo en neurocirugía.

Una de las modalidades de imágenes utilizadas en la investigación fundamental y clínica, es la Resonancia Magnética de Difusión (MRI) (Le Bihan et al. (2001), Le Bihan and Lima (2015)), que hace posible el análisis de los tractos de fibras de la materia blanca. Gracias a este método, se ha avanzado en el estudio de enfermedades como la esclerosis múltiple (Filippi et al. (2016)), la esquizofrenia (Yao et al. (2017)), el Alzheimer (Frisoni et al. (2010)) y tumores cerebrales debido a la detección de cambios en la difusión global o en las fibras hipotéticamente afectadas.

Es posible procesar los datos de imágenes de MRI de difusión con una técnica llamada tractografía de fibras. La tractografía utiliza la información de las imágenes adquiridas mediante MRI para reconstruir las principales trayectorias de las fibras cerebrales mediante su secuencia de coordenadas en el espacio tridimensional. Los tractos de fibras producidas por este método conectan zonas distantes y también cercanas del cerebro. La tractografía permite realizar análisis de la estructura de la materia blanca en diferentes poblaciones de sujetos. Para ello existen técnicas como la segmentación o el clustering de fibras.

El tema principal de la presente tesis es el clustering de fibras, que consiste en agrupar fibras, con similar forma, trayectoria y posición, en fascículos, partiendo de las fibras en 3D de una tractografía (O'Donnell et al. (2006), Ros et al. (2013)). Existen numerosos métodos de clustering y grupos de investigación tratando de mejorar el tiempo de cómputo y la calidad de los clústeres.

A pesar de las mejoras en el hardware de los equipos (más memoria RAM, mejor procesador, más capacidad de almacenamiento), cada vez existe un mayor volumen de imágenes a analizar con una gran resolución y los datos tractográficos pueden llegar a ocupar más de 7G de memoria RAM durante su análisis. Por ese motivo, un importante reto en el campo, es el tratamiento de grandes volúmenes de datos de tractografías, manteniendo la calidad y un tiempo clínicamente aceptable. Adicionalmente, el gran tamaño de los datos de tractografías y el tiempo de cómputo imposibilitan la visualización e interacción mediante herramientas de visualización.

En esta tesis se propone desarrollar algoritmos para realizar de manera eficiente la tarea de clustering sobre fibras, mejorando el tiempo de cómputo con respecto a los algoritmos existentes en el estado del arte y manteniendo la calidad de los resultados.

1.2. HIPÓTESIS

Es posible mejorar el tiempo de cómputo en la tarea de clustering de fibras de la materia blanca del cerebro con respecto a los demás algoritmos existentes mediante el diseño y la implementación de algoritmos de clustering paralelos sin una pérdida significativa de calidad en los clústeres. Esto conlleva a un mejor rendimiento en los software de visualización en los que se implementa.

1.3. OBJETIVOS

1.3.1. OBJETIVO GENERAL

Diseño, implementación y evaluación de algoritmos paralelos para realizar eficientemente la tarea de clustering de fibras en fascículos.

1.3.2. OBJETIVOS ESPECÍFICOS

- Diseñar e implementar un algoritmo paralelo eficiente para clustering de fibras.
- Evaluación del rendimiento mediante experimentación, realizando una comparativa en tiempo con otros algoritmos existentes y con diferentes conjuntos de datos.

1.4. ESTRUCTURA DE LA TESIS

La presente tesis se estructura de este modo:

- Capítulo 1: se muestra una introducción del trabajo, así como la hipótesis, objetivo general y objetivos específicos.
- Capítulo 2: se exponen los conceptos teóricos, el trabajo relacionado con el contenido de esta tesis y el estado del arte del clustering de fibras de la materia blanca y la visualización de las mismas.
- Capítulo 3: se describe la base de datos utilizada y los métodos implementados de clustering y segmentación de fibras de la materia blanca.
- Capítulo 4: se presentan los resultados de la investigación.
- Capítulo 5: se enuncian las conclusiones, el trabajo futuro y las publicaciones obtenidas con la investigación.

Capítulo 2

ESTADO DEL ARTE

2.1. INTRODUCCIÓN

En el presente capítulo se realiza una breve descripción de los principales trabajos relacionados con algoritmos de clustering aplicados a la segmentación de fibras y métodos de visualización, interacción y compresión de fibras. Previamente a la descripción del trabajo relacionado, se describen brevemente algunos de los conceptos más importantes para comprender mejor el trabajo de la presente tesis.

2.2. CONCEPTOS TEÓRICOS

2.2.1. Imágenes de Resonancia Magnética de Difusión

La Resonancia Magnética de Difusión es una técnica no invasiva que permite reconstruir la trayectoria de las fibras mediante la detección de la difusión de las moléculas de agua (Merboldt et al. (1985), Le Bihan et al. (2001)). La difusión de las moléculas de agua permite generar contraste en las imágenes debido a que no es una difusión libre, sino que las moléculas interactúan con obstáculos, como las moléculas o membranas del tejido (Haacke et al. (1999)). Para poder adquirir las imágenes, se utilizan escáneres de resonancia magnética, que aplican campos magnéticos, ondas de radio y gradientes en el tejido vivo.

Los datos de tractografías utilizados en el presente trabajo, fueron calculados a partir de Imágenes de Resonancia Magnética de Difusión del cerebro, y contienen una representación de la anatomía de las fibras de la materia blanca (WM).

En la Figura 2.1 se puede ver un ejemplo de Imagen de Resonancia Magnética



Figura 2.1: Imagen de Resonancia Magnética Anatómica (contraste T1). Las zonas más oscuras representan una mayor concentración de agua y las más claras representan el tejido. Es una imagen de diferentes contrastes entre la materia blanca, la materia gris y el fluido cerebroespinal. Fuente: Haacke et al. (1999).

anatómica.

2.2.2. Tractografía

Una tractografía es una técnica de modelado que, a partir de datos extraídos de imágenes de resonancia magnética por difusión, devuelve una representación 3D de los principales tractos de fibras (Basser et al. (2000)). Cada una de las fibras representadas por un conjunto de puntos en el espacio 3D recibe el nombre de *streamline*. En el presente trabajo se utilizan tractografías de tractos neurales, en concreto, cada trayectoria de las fibras de la materia blanca del cerebro es representada mediante una secuencia de coordenadas en el espacio 3D.

En la Figura 2.2 se puede ver un ejemplo de una reconstrucción 3D de las fibras de una tractografía del cerebro completo.

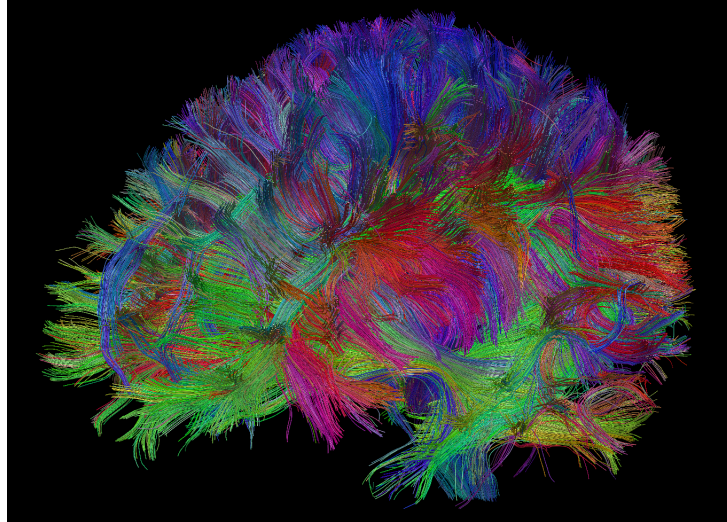


Figura 2.2: Representación 3D de las fibras de una tractografía (cerebro completo). Esta tractografía se creó a partir de un modelo llamado deconvolución esférica limitada utilizando datos dMRI de una MRI de alta sensibilidad (el escáner Connectome) en el centro *Martinos Center for Brain Imaging*, Hospital General de Massachusetts, en Charlestown, Boston, Massachusetts. Fuente: Harvard (2018).

2.2.3. Clustering y segmentación de fibras

Clustering de fibras: Se trata de una técnica que consiste en descubrir grupos de fibras (fascículos) en una tractografía. Para formar grupos, las fibras tienen que compartir una serie de características, como posición, proximidad y longitud. Los algoritmos de clustering no son supervisados, y forman grupos con la totalidad de las fibras de la tractografía (Jin et al. (2014), Visser et al. (2011), Guevara et al. (2011a)).

Segmentación de fibras: Se trata de una técnica que consiste en agrupar fibras en fascículos conocidos en una tractografía, utilizando como guía un atlas de fibras (Guevara et al. (2012a), Labra et al. (2017)). El atlas de fibras está formado por los fascículos conocidos y previamente estudiados de múltiples sujetos. Los algoritmos de segmentación comparan las fibras de la tractografía de un sujeto, con las fibras del atlas, creando agrupaciones en base a los fascículos de dicho atlas. A diferencia del clustering de fibras, la segmentación de fibras no descubre nuevos fascículos.

2.3. TRABAJO RELACIONADO

Para la clasificación de fibras en fascículos conocidos se han propuesto un gran número de algoritmos.

Entre ellos se destaca el algoritmo robusto de clustering de fibras propuesto por O'Donnell et al. (2006). Este está basado en un clustering espectral, aplicado a una muestra de los datos. Utilizó el método de Runge-Kutta (Ascher and Petzold (1998)) para realizar la tractografía. El procedimiento de clustering de fibras se realizó en función de la similitud entre pares, teniendo en cuenta su forma y distribución espacial.

Más adelante, se han implementado otros algoritmos destacables para la segmentación automática de fibras en fascículos conocidos. Por ejemplo, existe un método basado en atlas multisujeto de fascículos de fibras conocidas (Guevara et al. (2012b)). Este atlas es un modelo de la organización de la materia blanca del cerebro. Se creó utilizando dos niveles de estrategias de clustering: intrasujeto (en forma individual, obtiene una buena representación de la variabilidad de la forma y localización) e intersujeto (clústeres con correspondencia entre los sujetos). Los fascículos del atlas fueron creados a partir del etiquetado manual de los clústeres intersujeto, tomando como referencia un conjunto de individuos para tratar de conseguir fascículos genéricos para la mayoría de la población. Existe otro método de clustering guiado por atlas, que realiza una rápida y consistente extracción de fascículos (Ros et al. (2013)) a través de un framework que utiliza análisis jerárquico de clústeres para explotar la redundancia en conjuntos de datos de gran tamaño (hasta un millón de streamlines).

Es necesario tener en cuenta que se generan muchos streamlines incorrectos en las tractografías, lo que puede conllevar a una mala clasificación. Para extraer tractos consistentes con una anatomía conocida, se implementó un algoritmo basado en fusión de etiquetas (Jin et al. (2014)) que mapea múltiples etiquetas (puestas manualmente en un atlas) en un nuevo conjunto de datos.

2.3.1. Clustering exploratorio

No solo se han implementado algoritmos para agrupar fibras en fascículos conocidos, sino que también se propusieron métodos de clustering que devuelven grupos de fibras que representan zonas relevantes en la materia blanca (Visser et al. (2011), Guevara et al. (2011a), Garyfallidis et al. (2012)). Una gran ventaja de dichos métodos es que se utilizan con grandes conjuntos de datos de tractografías. En el trabajo propuesto por Guevara et al. (2011a) se realiza el proceso de clustering de los vóxeles de la materia blanca en lugar de fibras, es capaz de comprimir (reducir la dimensionalidad) del orden de millones de fibras a miles, lo que supone una gran ventaja para trabajar con grandes volúmenes de datos de tractografías. Otro importante trabajo que realiza clustering con tractografías de gran tamaño es QuickBundles (Garyfallidis et al. (2012)), un algoritmo de clustering no supervisado que agrupa las fibras en clústeres una única vez, sin recalcular los clústeres, y de forma bastante rápida.

Un algoritmo de clustering interesante, es el implementado por Sanchez et al. (2018) en el que se agrupan las fibras en función de su distribución de puntos, en los cuales las fibras deben de ser muy similares entre sí, de gran relevancia para el análisis de fascículos de asociación cortos.

Uno de los grandes objetivos de este tipo de algoritmos de clustering de tipo exploratorio, es representar la estructura de las principales vías de la materia blanca, reduciendo la dimensionalidad, con el objetivo de poder aplicar subsecuentemente otros análisis sobre estos clústeres. Otro objetivo de los métodos mencionados anteriormente, es su utilización en herramientas de visualización e interacción con las diferentes regiones del cerebro, también con fines exploratorios y para segmentación manual o semi-automática de las fibras. Se trata siempre de mejorar el proceso de clustering de fibras en fascículos, buscando alcanzar una representación que permita posteriormente un análisis del punto de vista anatómico, y en un tiempo clínicamente aceptable.

Existen herramientas que tratan de reducir el tiempo de visualización utilizando técnicas de OpenGL moderno como shaders y Primitive Restart, donde se reduce el número de llamadas a la hora de dibujar fibras (Bonometti et al. (2015)). Además, no solo es importante reducir el tiempo de visualización, sino también el de interacción con los fascículos (Rheault et al. (2017)). En este trabajo se propone una técnica de interacción progresiva basada en segmentos y puntos para la compresión de tractogramas, lo que, combinado con un proceso de carga eficiente, le permite tratar con millones de fibras. Una herramienta innovadora es iFiber (Guevara et al. (2015)), cuyo objetivo es la visualización de datos de tractografías en dispositivos móviles Android, basado en el mismo "shader" desarrollado en Bonometti et al. (2015).

Recientemente, se ha publicado una nueva herramienta de código libre, AFQ-Browser (Yeatman et al. (2018)) para visualizar y analizar imágenes de resonancia magnética (MRI). Una gran ventaja de este método es que funciona en cualquier navegador web, facilitando la transparencia y la compartición de datos entre distintas comunidades.

De todos los trabajos mencionados anteriormente, algunos tienen una mayor relevancia para la presente tesis, por lo que es necesario analizarlos más detalladamente. A continuación, se describen los trabajos relacionados más importantes, la sección 2.3 contiene los trabajos relacionados con el clustering de fibras y en la sección 2.4 se detallan algunas herramientas de visualización y análisis de la materia blanca.

2.4. CLUSTERING DE FIBRAS

En esta sección, se describen algunos de los trabajos más relevantes relacionados con el clustering de fibras del cerebro del tipo exploratorio, es decir, que sólo buscan agrupar fibras similares y no en fascículos anatómicos.

2.4.1. Clustering sobre conjuntos de datos fibras masivos

Es importante destacar un último trabajo en la categoría de clustering relacionado con el presente proyecto, se trata de un método implementado por Guevara et al. (2011a), que realiza clustering sobre conjuntos de datos con millones de fibras. El método propuesto consiste en una serie de algoritmos que dan lugar a un proceso robusto de clustering jerárquico que trabaja con millones de fibras, devolviendo un conjunto de miles de clústeres homogéneos. A continuación se describe brevemente cada una de las etapas del clustering jerárquico:

- Descomposición jerárquica: El primer paso consiste en realizar segmentación utilizando máscaras de los hemisferios y el cerebelo dadas por el software Brain-Visa (Mangin et al. (1996)).
- Segmentación basada en longitud: A partir del resultado anterior, se dividen las fibras con tamaño similar en grupos.
- Clustering basado en vóxeles: Cada grupo se subdivide en grupos de fibras que conectan vóxeles para generar una representación de la materia blanca. El punto clave de esta representación basada en vóxeles, es que se agrupan cuando son atravesados por muchas fibras similares, creando una máscara más robusta de los fascículos subyacentes.
- Clustering basado en extremidades: Las fibras de los grupos obtenidas, se subdividen en subgrupos de fibras con extremidades muy cercanas. Se realiza clustering para detectar regiones 3D en las extremidades de los tractos con alta densidad. Cada par de dichas regiones forman un fascículo homogéneo.
- Finalmente, se fusionan los fascículos con geometría similar, utilizando un clustering jerárquico.

A pesar de presentar una buena calidad en los clústeres, este método presenta las siguientes desventajas:

- Tiempo de procesamiento: El tiempo de ejecución se demora entre una y dos horas. Esto puede ser una gran desventaja si se trata de procesar múltiples sujetos o en el caso de que se utilice el algoritmo en un software de visualización.
- Requiere información anatómica: El método depende de utilizar máscaras del cerebro obtenidas del software BrainVisa.
- Genera una gran cantidad de archivos.
- Depende de la densidad de las fibras.

En la Figura 2.3 se muestra un esquema de todo el proceso de clustering. La aplicación del método de clustering jerárquico propuesto da lugar a una serie de fascículos bien diferenciados y de gran calidad (ver Figura 2.4).

2.4.2. QuickBundles

Otro trabajo destacado es un algoritmo llamado QuickBundles (QB), basado en clustering no supervisado propuesto por Garyfallidis et al. (2012), cuyo objetivo es agrupar fibras utilizando grandes conjuntos de datos y en un tiempo clínicamente aceptable.

La principal característica es que, los clústeres se van creando a medida que se comparan las fibras, es decir, a diferencia de otros métodos como K-means, los clústeres se crean una única vez y no se vuelven a recalcular. Adicionalmente, se introduce una nueva medida, la adyacencia de fascículos, Bundle Adjacency (BA) que ayuda a encontrar similitudes entre diferentes sujetos, seleccionando los 100 clústeres más grandes devueltos por QB (los clústeres con un mayor número de fibras) y comparándolos con BA.

A continuación, se describe el método utilizado, donde s y t son dos streamlines, K es el número de puntos de los streamlines s , y s^F y t^F son los dos streamlines invertidos:

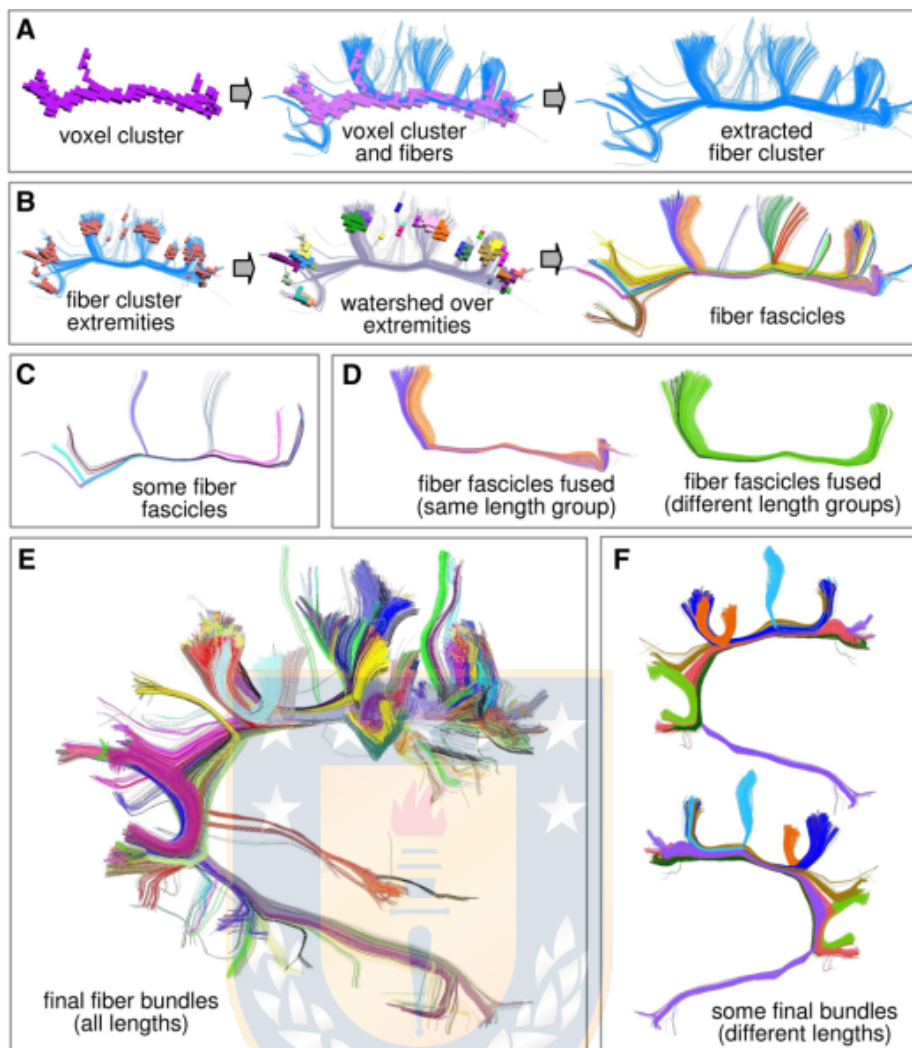


Figura 2.3: Esquema general del método de segmentación de fibras. Paso 1 (STEP 1): Descomposición jerárquica: El conjunto de fibras se segmenta en cuatro subconjuntos principales. Paso 2 (STEP 2): Segmentación basada en longitud: Se separan las fibras de cada subconjunto en grupos que contienen fibras de tamaño similar. Paso 3 (STEP 3): Clustering basado en vóxeles: Se subdividen las fibras de cada grupo anterior a través de la parcelación de la materia blanca basada en la conectividad de las fibras. Paso 4 (STEP 4): Clustering basado en extremidades: Las fibras de los clústeres se dividen en fascículos utilizando las extremidades de las fibras. Paso 5 (STEP 5): Fusión de fascículos: Los fascículos que tienen una geometría similar se fusionan. Fuente: Guevara et al. (2011a).

En primer lugar, se muestran los conceptos principales del algoritmo QB:

- Los streamlines son un conjunto de puntos en R^3 .

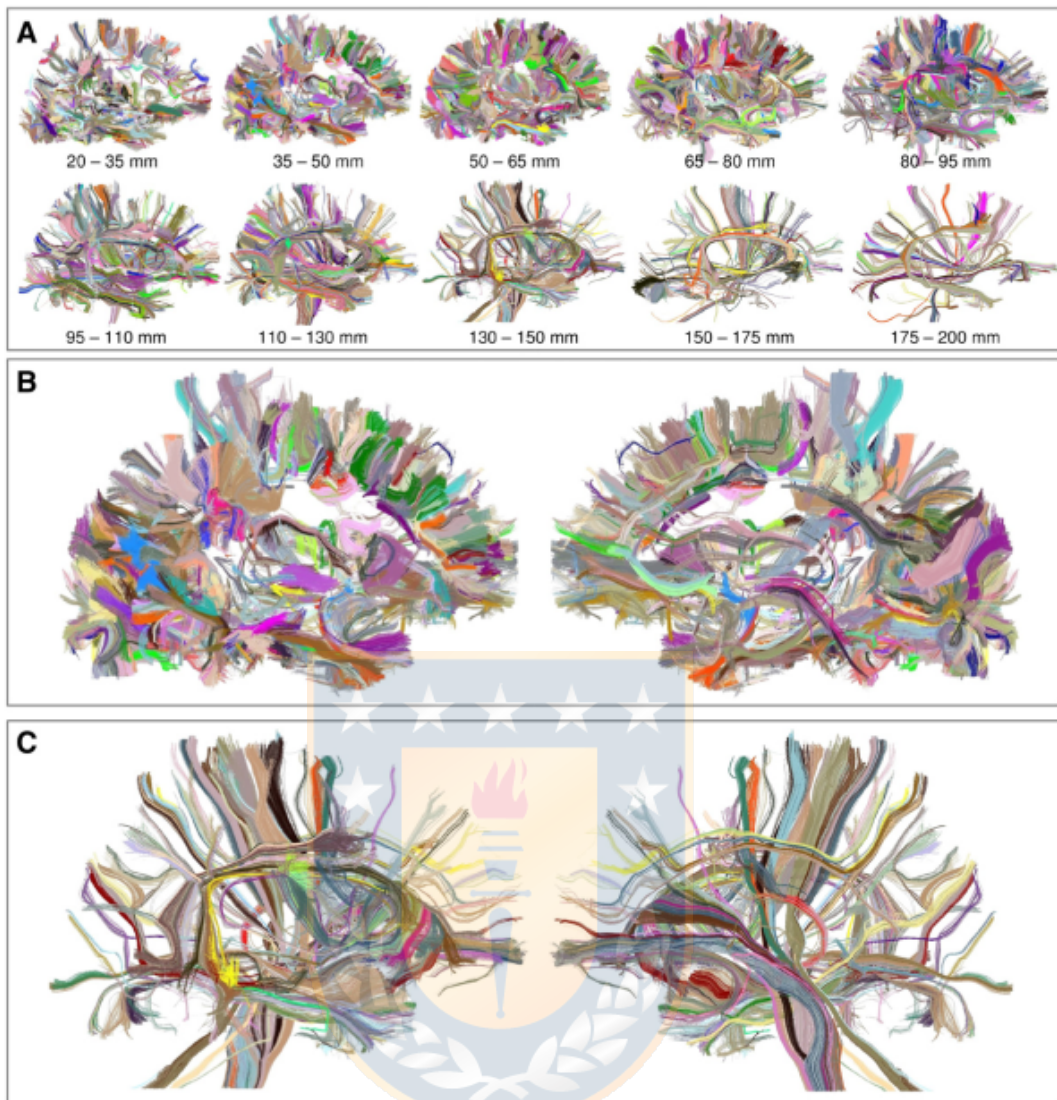


Figura 2.4: Fascículos resultantes para el hemisferio izquierdo de un adulto. Cada fascículo tiene un color aleatorio asignado. A: Vista externa del conjunto completo de resultados. Cada fascículo está separado en 10 grupos de diferente longitud (para facilitar la visualización). B: Vista exterior e interior de un grupo de fibras cortas. C: Vistas exterior e interior del grupo de fibras largas. Fuente: Guevara et al. (2011a).

- Los streamlines se dividen en un número de puntos equidistantes (generalmente 12).
- QB utiliza como medida de distancia la *Mínimum Average Direct Flip (MDF)*.

- Distancia de un streamline a otro:

$$d_{direct}(s, t) = d(s, t) = \frac{1}{K} \sum_{i=1}^K |s_i + t_i|$$

- Distancia de un streamline a otro donde alguno está invertido:

$$d_{flipped}(s, t) = d(s, t^F) = d(s^F, t)$$

- Distancia MDF:

$$MDF(s, t) = \min(d_{direct}(s, t), d_{flipped}(s, t))$$

- La información de los clústeres se almacenan en nodos de clúster. Un nodo de clúster se define como $c = (l, h, n)$, donde l son los índices de los streamlines de cada clúster, h es la suma de los streamlines del clúster y n es el número de streamlines en el clúster.
- QB también almacena, a modo de resumen, el streamline centroide v , que se calcula: $v = h/n$.

A continuación, se enumeran los pasos del algoritmo:

1. Se selecciona el primer streamline s_1 y se añade al primer clúster.
2. Para cada streamline s_i restante repetir hasta que no queden streamlines sin agrupar:
 - a) Se calcula la MDF entre el streamline s_i y los centroides de los clústeres existentes.
 - b) Si existen distancias MDF menores al umbral establecido, seleccionar el menor valor y añadir el streamline al clúster correspondiente. Actualizar el nodo de clúster $c = (l, h, n)$.
 - c) Si las distancias MDF superan el umbral, crear un nuevo clúster y añadirle el streamline.

La complejidad de QB en el mejor caso es lineal $O(N)$ en el número de streamlines. En el peor caso es $O(N^2)$ y sucede cuando todos los clústeres contienen un único streamline.

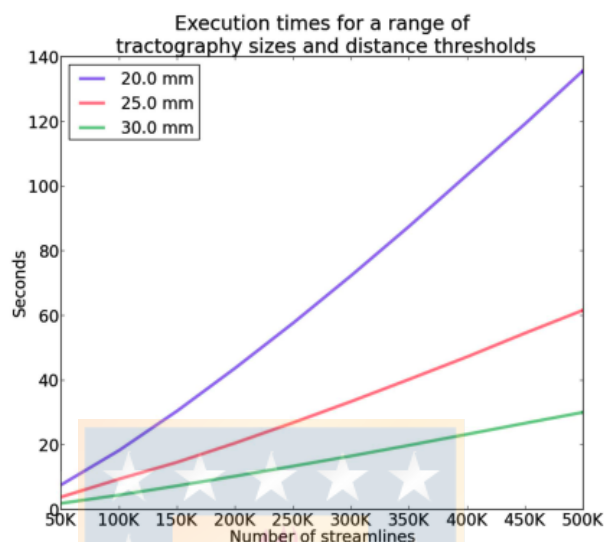


Figura 2.5: Complejidad temporal del algoritmo QB. Eje x: Número de streamlines. Eje y: segundos que tarda QB en procesar los streamlines según los diferentes umbrales de distancia. Fuente: Garyfallidis et al. (2012).

En la gráfica de la Figura 2.5 se puede observar la complejidad temporal del algoritmo QB en función del número de streamlines. A medida que aumenta el número de streamlines, aumenta el tiempo de procesamiento de forma casi proporcional al tamaño de los datos.

Otro detalle importante es el valor de umbral de distancia, a medida que este valor aumenta, disminuye el tiempo, debido a que los clústeres aceptan más cantidad de streamlines y, por lo tanto, se crean menos clústeres.

En la Figura 2.6 se puede apreciar cómo varía el número de clústeres en función del tamaño del umbral seleccionado.

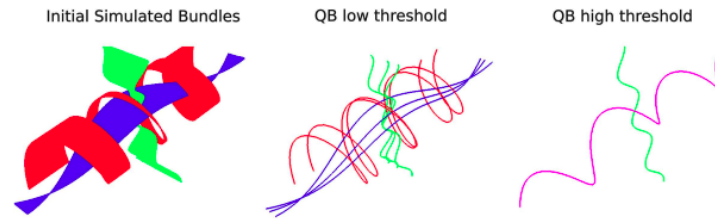


Figura 2.6: Variación del número de clústeres en función del umbral de distancia mínima. En la imagen de la derecha se pueden observar los clústeres originales (verde, rojo y azul) con todas las fibras. Las imágenes del centro y derecha son el resultado de aplicar umbrales de 8 y 1 respectivamente. Cuanto menor es el umbral de distancia mínima, menos fibras acepta en los clústeres. Fuente: Garyfallidis et al. (2012).

2.4.3. Clustering basado en K-means sobre puntos de las fibras

En el proyecto realizado por Sanchez et al. (2018) se propone un algoritmo que mejora el rendimiento del proceso de clustering de fibras a partir de conjuntos de datos extraídos de tractografías.

El método implementado se divide en tres partes (ver Figura 2.7):

- (i) Se parte con las fibras (o streamlines) remuestreadas en un conjunto fijo de puntos equidistantes. Las fibras tienen formas 3D en cualquier orientación en el espacio. Además están almacenadas en la memoria de forma aleatoria, donde dos fibras con la misma orientación y posición, pueden estar almacenadas de manera ordenada (similar entre ambas) o invertida (flipped). El primer paso consiste en aplicar el algoritmo Minibatch K-means (Sculley (2010)) en paralelo sobre algunos puntos definidos de los streamlines. Esto agrupa los puntos de los diferentes streamlines según la proximidad entre ellos.
- (ii) Una vez obtenidos los clústeres de puntos, se agrupan las fibras que tengan todas las etiquetas iguales para los puntos donde se aplicó el clustering del paso I. Se crea un diccionario en el que, cada clave es el conjunto de etiquetas de los clústeres de puntos que pertenecen a un streamline y, el valor, es el conjunto de streamlines que comparten los mismos clústeres de puntos, es decir, comparten la clave. De esta manera se obtienen grupos de fibras, provenientes de los clústeres de puntos iguales.

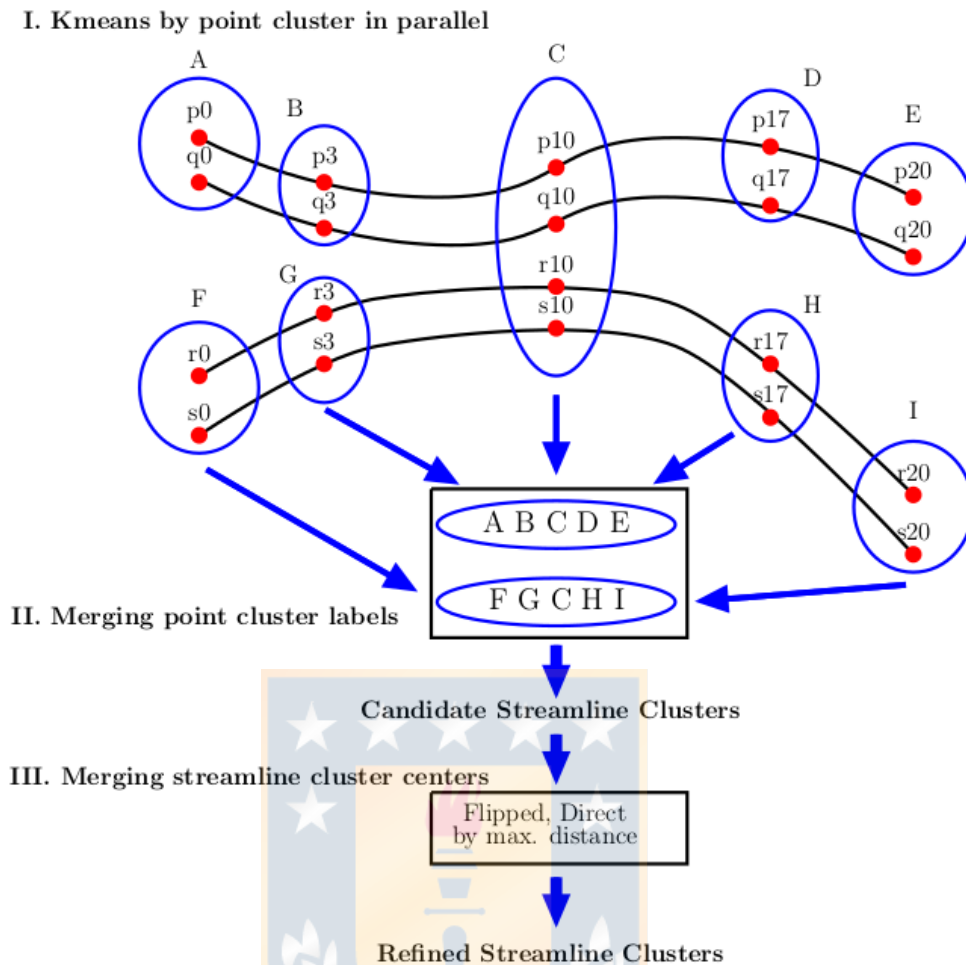


Figura 2.7: Pasos para realizar clustering: I. Se aplica el algoritmo Minibatch K-means en paralelo (multiprocesador) sobre los puntos seleccionados, obteniendo clústeres de puntos. II. Fusión de etiquetas: Los streamlines se identifican con una lista de etiquetas de los clúster a los que pertenece cada uno de sus puntos, formando así grupos de streamlines. III. Fusión de centros de clúster: Se fusionan los clústeres de los grupos de streamline que comparten el clúster del punto medio, a partir de sus centroides, después se añaden los streamlines cuya distancia no supere el umbral máximo establecido. Fuente: Sanchez et al. (2018).

- (III) Por último, para reducir el número final de clústeres de fibras, se fusionan los grupos de fibras obtenidos en el paso anterior. Para ello, se analizan los grupos de fibras cuyos streamlines provengan del mismo clúster del punto central. Para garantizar que sólo se fusionan clústeres similares, los streamlines deben cumplir con otro criterio: que la distancia entre ellos no supere el umbral máximo

establecido. Es posible que los streamlines en el conjunto de datos no se encuentren todos con la misma orientación, es decir, pueden estar invertidos unos con respecto a otros. Por ese motivo, es necesario calcular la distancia con el streamline en su orientación actual y con la orientación invertida. A continuación, se muestra el método utilizado para calcular la distancia, donde a y b son los streamlines, a_i y b_i representan cada punto del streamline, donde $i > 0$, K es el número de puntos en un streamline y a^F y b^F son los streamlines invertidos:

- a) Distancia Euclídea entre dos streamlines considerando que ambas fibras están ordenadas de la misma forma (ver 2.1).

$$d_E(a, b) = \|a_i - b_i\| = \left(\sum_{i=1}^K (a_i - b_i)^2 \right)^{\frac{1}{2}} \quad (2.1)$$

- b) Distancia entre dos streamlines, considerando que una de ellas está invertida en el espacio con respecto a la otra (ver 2.2).

$$d_{Eflip}(a, b) = d_E(a, b^F) = d_E(a^F, b) \quad (2.2)$$

- c) Finalmente se calcula la distancia mínima entre las dos anteriores (ver 2.3).

$$d_{ME} = \min(\max(d_E(a, b)), \max(d_{Eflip}(a, b))) \quad (2.3)$$

Para la experimentación, se utilizaron datos extraídos de la base de datos ARCHI (Schmitt et al. (2012b)), en concreto, el hemisferio izquierdo de un sujeto con 258.382 fibras representadas mediante 21 puntos (coordenadas 3D) equidistantes. El Mini-batch K-means se aplica sobre los puntos 1 (extremo), 4 (intermedio), 11 (central), 17 (intermedio) y 21 (extremo). Después de aplicar el paso II, se forman 41.764 clústeres de streamline, a los que se les descarta los clústeres con un único streamline, quedando 23.660 clústeres de streamline. Finalmente, en el paso III, quedan 16.497 después de mezclar los centros de los clúster (con el umbral máximo de distancia entre centroides establecido en 10mm). En la Figura 2.8 se muestran los centroides de los resultados obtenidos.

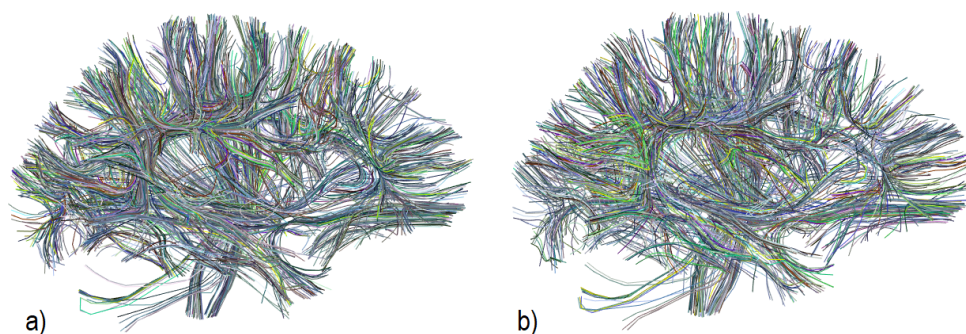


Figura 2.8: Centroides de los clústeres de los streamlines. En la imagen a) se muestran los streamlines correspondientes a centroides de los clústeres después de aplicar el paso 2 del algoritmo. En la imagen b) se muestran los streamlines correspondientes a los centroides después de aplicar el paso 3. Fuente: Sanchez et al. (2018).

2.5. VISUALIZACIÓN E INTERACCIÓN

En esta sección, se describen algunas herramientas importantes para visualizar fibras e interactuar con los fascículos.

2.5.1. Visualización con OpenGL moderno

La aplicación propuesta por Bonometti et al. (2015), utiliza OpenGL moderno para visualizar tractografías de manera rápida e interactiva con el objetivo de mejorar el rendimiento y la visualización de fibras cerebrales. Se realizan pruebas en GPU debido a su buen rendimiento en gráficos con OpenGL.

Se realizan dos implementaciones, Renderizado Directo (RD) que dibuja cada fibra de forma separada y tiene pequeños problemas de rendimiento durante el dibujado y Primitive Restart (PR), técnica de optimización de OpenGL que reduce el número de llamadas a la hora de dibujar las fibras.

En la Figura 2.9 se muestra la diferencia en la representación de fibras utilizando colores planos (arriba) o el algoritmo de iluminación de "Phong". En la Figura 2.10 se muestra la comparativa en tiempo entre DR (Direct Rendering) y PR (Primitive Restart) utilizando diferentes GPUs y diferentes tamaños de los conjuntos de datos.

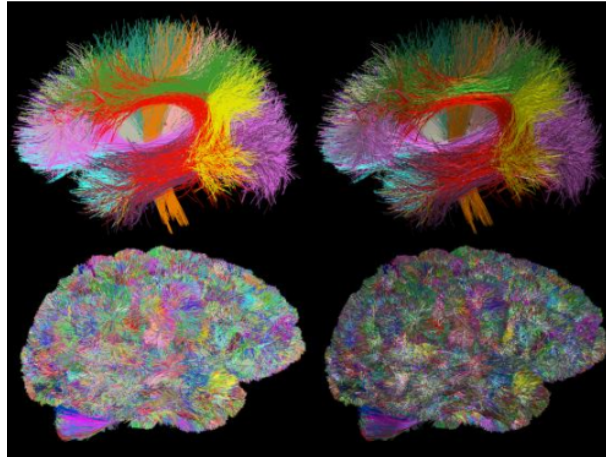


Figura 2.9: Comparativa entre colores planos y el algoritmo de iluminación de Phong para un atlas de fascículos de la materia blanca (Bonometti et al. (2015)) y un conjunto de datos de clustering. Fuente: Guevara et al. (2011a).

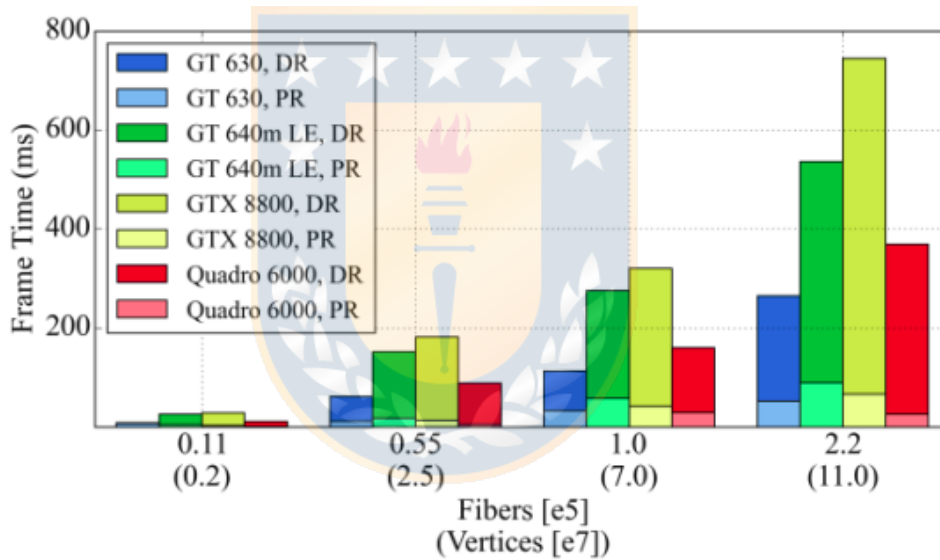


Figura 2.10: Comparativa entre Direct Rendering y Primitive Restart. En la gráfica se muestran los tiempos de frame para las diferentes plataformas y conjuntos de datos de tractografías. Fuente: Bonometti et al. (2015).

2.5.2. Interacción y visualización de grandes volúmenes de datos

Uno de los problemas más importantes relacionados con la carga y la visualización es el tratamiento de grandes volúmenes de datos extraídos de tractografías. Por cada una de ellas se producen millones de streamlines. Por ejemplo, una tractografía con

paso igual a 0.5mm (espacio entre los puntos consecutivos de un streamline) y con 700 mil streamlines ocuparía aproximadamente 1GB en el disco y más de 7GB en memoria RAM.

El gran tamaño de los datos de tractografías también suponen un problema para su transferencia (debido a limitaciones con el ancho de banda) y su visualización. Es posible comprimir los datos de tractografías mediante diferentes técnicas, como, por ejemplo, eliminando puntos de los streamlines, pero también surgen problemas debido a la pérdida de información. En el trabajo realizado por Rheault et al. (2017) se proponen las siguientes soluciones:

- Procedimiento de carga más eficiente para mejorar y acelerar la visualización, reduciendo el uso de memoria.
- Técnicas de interacción más robustas para comprimir tractogramas.

Para mejorar la carga y visualización, se utiliza el proceso de linearización de Presseau et al. (2015) durante la carga. Dicho proceso consiste en la eliminación de puntos colineales de los streamlines durante el tiempo de carga, lo que reduce el tiempo de renderizado del tractograma y el tamaño del fichero.

El proceso de linearización se realiza comprobando punto por punto en cada streamline y utilizando un parámetro, el umbral de error máximo, para evitar perder información. A medida que se comprueban puntos, es necesario comprobar también los puntos anteriores para evitar que el streamline resultante se desvíe. A esta comprobación se le llama *backward verification* (verificación hacia atrás) y ralentiza el proceso de carga.

El aporte realizado, es la introducción de un nuevo parámetro, la distancia de linearización máxima (MLD) que evita verificaciones hacia atrás demasiado largas y con ello se acelera el proceso de carga de millones de fibras sin perder mucha información.

El otro aporte de Rheault et al. (2017) es la interacción robusta con streamlines comprimidos. Para la selección de los diferentes bundles, se utilizan ROIs, que son figuras geométricas (cajas o esferas) que se sitúan en ciertas regiones de interés en la aplicación gráfica. El objetivo es identificar los streamlines comprimidos de cada región. Los procedimientos son los siguientes:

- Basado en puntos: Identifica streamlines a partir de puntos dentro de la región. El problema, es que con streamlines comprimidos se pierde mucha información.
- Basado en segmentos: Identifica streamlines a partir de segmentos que intersecan o están en la región de interés. El problema es generar millones de segmentos, donde el tiempo de cómputo es muy elevado.

El aporte es una aproximación progresiva que combina los dos procedimientos anteriores. Los pasos son los siguientes:

1. Se identifican los puntos dentro del ROI mediante un octree.
2. Se buscan los streamlines que faltan ampliando la región del vecindario en función de la MLD. Esto permite identificar la mayoría de streamlines con poca pérdida de información. La ampliación del vecindario también podría ser la longitud del segmento máximo o la longitud media del segmento en el conjunto de datos.

En la Tabla 2.1, se muestra una comparativa en tiempo de compresión utilizando diferentes valores del nuevo parámetro, MDL. Se puede apreciar que, sin utilizar la restricción de MDL, el tiempo de compresión es mayor.

Como se acaba de describir en este capítulo, actualmente existen múltiples programas de visualización e interacción con los diferentes fascículos de la materia blanca del cerebro. Algunos permiten visualizar las regiones de la materia blanca resultantes de aplicar clustering exploratorio e interactuar con las fibras, otros con fascículos segmentados a partir de atlas con una anatomía conocida. Dichos programas de visualización pueden servir como herramientas de apoyo para especialistas médicos o para investigadores en el campo de la neurociencia, incluso pueden llegar a ser necesarias herramientas de visualización en tiempo real.

MET (mm)	MLD = 5 mm (ms)	MLD = 10 mm (ms)	MLD = 25 mm (ms)	No constraint (ms)
0.1	48,526	52,755	52,943	53,198
0.2	53,038	79,843	82,342	83,982
0.3	55,137	88,069	98,313	99,524
0.4	55,517	99,513	117,710	119,071
0.5	55,771	119,512	154,080	157,731

The dataset used for the table was the tractogram with 2 millions deterministic streamlines.

Cuadro 2.1: Comparativa del tiempo de compresión en función del MDL. Se mide el tiempo con diferentes valores MDL, desde una linearización no restringida hasta una linearización restringida hasta 5mm y con diferentes valores MET (Maximum Error Threshold), que es un umbral máximo de distancia para descartar puntos en el proceso de linearización. Fuente: Rheault et al. (2017).

2.6. DISCUSIÓN

La Resonancia Magnética de Difusión (dMRI) es una técnica no invasiva que permite la reconstrucción de las fibras mediante el movimiento de las moléculas del agua. Mediante la técnica conocida como tractografía, se extraen datos de las imágenes de resonancia magnética de difusión y permite realizar una reconstrucción 3D de los tractos de la materia blanca del cerebro. Cada fibra, se representa en el espacio 3D por un conjunto de puntos, se denomina *streamline*.

Para poder realizar la clasificación de las fibras en fascículos conocidos se han propuesto muchos algoritmos. El algoritmo propuesto por O'Donnell et al. (2006) basado en un clustering espectral, aplicado a una muestra de los datos utilizando regiones de interés (ROIs). Utilizó la similitud entre pares teniendo en cuenta su distribución espacial y su forma.

En cuanto a la segmentación automática de fibras en fascículos conocidas, existe el método basado en atlas multisujeto de (Guevara et al. (2012b)). Este atlas es un modelo de organización de la materia blanca del cerebro creado utilizando dos niveles de estrategias de clustering intrasujeto e intersujeto. Otro método de clustering basado

en atlas es el del trabajo de Ros et al. (2013), que extrae rápidamente fascículos a través de un framework que implementa un análisis jerárquico de clústeres y explota conjuntos de gran tamaño (hasta un millón de streamlines).

Además de los algoritmos para agrupar fibras en fascículos bien conocidos, se han propuesto métodos de clustering que recuperan grupos de fibras que representan zonas importantes de la materia blanca (Visser et al. (2011), Guevara et al. (2011a), Garyfallidis et al. (2012)). La principal ventaja de estos métodos es que se utilizan con grandes conjuntos de datos de tractografías. En el trabajo propuesto por Guevara et al. (2011a), se llega a comprimir (reducción de la dimensionalidad) del orden de millones de fibras a miles, lo cual favorece el trabajar con grandes volúmenes de datos de tractografías. Otro de los trabajos más importantes de clustering con tractografías masivas es QuickBundles (Garyfallidis et al. (2012)), que utiliza un algoritmo de clustering no supervisado, y agrupa las fibras en clústeres una única vez, sin volver a calcular los clústeres, y lo realiza de manera bastante rápida.

El análisis de fascículos formados por fibras cortas es muy relevante, ya que son los más desconocidos y menos estudiados. Además, las fibras cortas suponen el 98 % de la totalidad de las fibras del cerebro, y conectan regiones de gran importancia. El 2 % de fibras restante, son las fibras largas, cuyos fascículos son los más estudiados y conocidos.

Un algoritmo de clustering muy interesante y de gran relevancia para el análisis de fascículos de asociación cortos es el realizado por Sanchez et al. (2018), en el que se agrupan las fibras según su distribución de puntos, (Guevara et al. (2017)), teniendo en cuenta que las fibras deben de ser muy similares entre sí.

El principal objetivo de este tipo de algoritmos de clustering, es la representación de los tractos de la materia blanca, reduciendo la dimensionalidad. De este modo se pueden aplicar análisis posteriores sobre los clústeres. Otro enfoque es su utilización en herramientas de visualización e interacción con las diferentes regiones del cerebro, para segmentación manual o semiautomática de las fibras. Gracias a ello se puede

obtener una buena representación anatómica y alcanzar un tiempo clínicamente permisible.

Algunos de dichos métodos devuelven clústeres con muy buena calidad visual, sin embargo presentan otras carencias, como por ejemplo el tiempo de ejecución (pueden demorarse entre horas y días), la dependencia de información anatómica y de software privativo. Otros métodos se centran en la velocidad de ejecución, como por ejemplo QuickBundles, sacrificando la calidad de los clústeres. El objetivo del método propuesto en esta tesis, es obtener clústeres de fibras de gran calidad, tratando de mejorar las deficiencias y desventajas de los métodos mencionados anteriormente. Se trata de buscar un equilibrio entre calidad, tiempo de ejecución, facilidad de uso e independencia de software privativo.

A continuación se destacan las ventajas del método propuesto con respecto a los métodos anteriores.

- Clústeres homogéneos: Al realizar clustering sobre determinados puntos de las fibras (incluyendo los puntos extremos), se garantiza que se agrupen fibras cercanas, con longitudes similares y con los extremos visualmente homogéneos con el clúster.
- Velocidad de ejecución: Se realiza clustering sobre pocos puntos en las fibras (cinco puntos) utilizando el método K-means, cuyo algoritmo tiene una complejidad lineal. Esto hace que el tiempo de ejecución sea muy bajo.
- Independencia de información anatómica: La única información que necesita el método, es la tractografía, a partir de ella el algoritmo crea grupos de fibras.
- Independencia de software privativo: No se requiere ningún tipo de software privativo para utilizar el método, únicamente se utilizan algunas librerías de Python.
- Escalable: Gracias a que el método está dividido en varias etapas, se podría mejorar en un futuro modificando cada una de ellas de manera independiente o agregando nuevas.

- Clústeres de fibras cortas: El algoritmo propuesto descubre tanto clústeres de fibras largas, como clústeres de fibras cortas. Estos últimos son muy importantes, ya que forman fascículos desconocidos que no están muy estudiados.

Para finalizar, existen herramientas que reducen el tiempo de visualización mediante técnicas de OpenGL, donde se reduce el número de llamadas para dibujar las fibras (Bonometti et al. (2015)). Aparte de reducir el tiempo para visualizar se busca la interacción con los fascículos (Rheault et al. (2017)). Otra herramienta es iFiber (Guevara et al. (2015)), con el objetivo de visualizar los datos de las tractografías en dispositivos Android.



Capítulo 3

MATERIALES Y MÉTODOS

3.1. INTRODUCCIÓN

En este capítulo se describen las características de la base de datos utilizada para el desarrollo de la tesis. Además se detallan tanto el método de segmentación de fibras realizado como el método de clustering implementados para esta tesis.

3.2. BASE DE DATOS

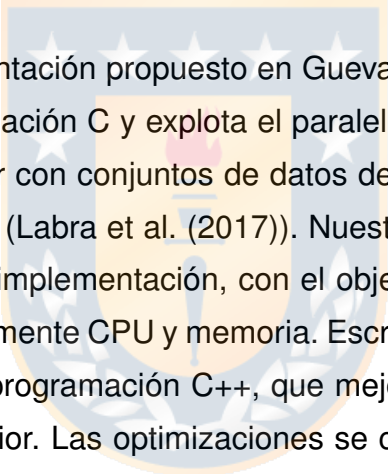
Se han utilizado sujetos sanos de la base de datos HARDI ARCHI (Schmitt et al. (2012a)), que contiene imágenes T1, HARDI y fMRI, adquiridas con secuencias de adquisición especiales en un escáner 3T MRI (Siemens, Erlangen). El protocolo de MRI incluyó un conjunto de datos de una sola capa HARDI SS-EPI a lo largo de 60 direcciones ponderadas en difusión, $b = 1500 \text{ s/mm}^2$ (70 cortes; matriz = 128x128; tamaño de vóxel = 1.71875x1.71875x1.7 mm).

Mediante el software BrainVISA/Connectomist-2.0 (ver <http://brainvisa.info/web/index.html>) los datos fueron preprocesados previamente. El modelo Q-ball analítico se aplicó para obtener campos de ODF en cada vóxel y se calculó una tractografía determinista de tipo streamline en toda la máscara cerebral basada en T1, con un paso de 0.2 mm y un ángulo de curvatura máximo de 30°.

3.3. MÉTODO DE SEGMENTACIÓN DE FIBRAS

Los métodos de segmentación de fascículos, que etiquetan los principales fascículos de fibras cerebrales, se implementan siguiendo dos estrategias principales.

Primero, se pueden usar varias regiones anatómicas de interés (ROI) para identificar las fibras que conectan dos o más regiones del cerebro. Este proceso se puede hacer automáticamente (Zhang et al. (2010); Wassermann et al. (2016)) o manualmente por un experto (Catani et al. (2002, 2012)). La segunda estrategia utiliza una métrica de distancia de fibras para identificar fibras similares, teniendo en cuenta su forma y posición relativa (O'Donnell and Westin (2007); Wassermann et al. (2010)). Estos métodos agregan información anatómica para identificar los fascículos de fibras con significado anatómico. Esta información, junto con la forma y la posición de la fibra para varios sujetos, se puede utilizar para crear un atlas de múltiples sujetos (Guevara et al. (2012a)). La segmentación basada en este tipo de atlas puede tratar la variabilidad que existe entre los sujetos y lograr buenos resultados en la clasificación de las fibras.



El algoritmo de segmentación propuesto en Guevara et al. (2012a) se implementó en el lenguaje de programación C y explota el paralelismo a nivel de subprocesos en múltiples CPUs para tratar con conjuntos de datos de tractografía masiva y lograr un corto tiempo de ejecución (Labra et al. (2017)). Nuestro propósito en este trabajo fue mejorar el algoritmo y su implementación, con el objetivo de reducir el uso de recursos informáticos, principalmente CPU y memoria. Escribimos la nueva implementación utilizando el lenguaje de programación C++, que mejora la modularidad y la extensibilidad de la versión anterior. Las optimizaciones se centraron en reducir la memoria utilizada para resultados temporales, mejorando la paralelización de tareas así como el algoritmo de descarte rápido de fascículos, y la clasificación de las fibras utilizando solo el fascículo más cercano.

El algoritmo optimizado mejora tanto el tiempo de ejecución como el uso de la memoria de su predecesor. Usando un conjunto de datos de 4.145.000 fibras con un atlas de 7.753 fibras (Guevara et al. (2017)) en una computadora con un procesador Intel i7-6700K y 32 GB de RAM, el nuevo algoritmo se ejecuta en 6 minutos, con una aceleración de 2,34 veces con respecto a la versión anterior. También reduce el uso de memoria en un factor de 0,79. Estas mejoras nos permiten realizar la segmentación

de conjuntos de datos muy grandes en una computadora de escritorio. Además, el nuevo algoritmo es más escalable, lo que permite la segmentación en conjuntos de datos de tractografía (y atlas) más masivos, en particular, los obtenidos de fibras de materia blanca superficial (SWM) con tractografía probabilística, que pueden alcanzar varios millones de fibras.

3.3.1. Atlas multisujeto

Los atlas multisujeto de SWM se construyeron en base a los centroides del clúster intrasujeto (Guevara et al. (2011b)) de una población de sujetos. El atlas LNAO-SWM79 se obtuvo de una base de datos HARDI de 79 sujetos (Schmitt et al. (2012a)), basado en una parcelación cortical para extraer las fibras que conectan dos regiones, seguido de un agrupamiento de fibras intra e intersujeto (100 fascículos, 7.753 centroides) (Guevara et al. (2017)).

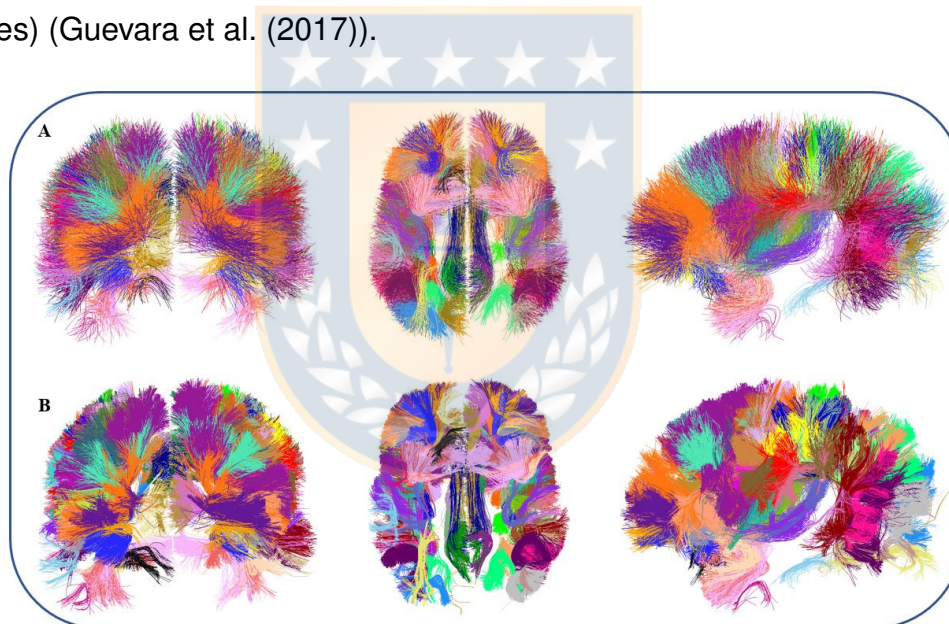


Figura 3.1: Atlas de fascículos de SWM y un sujeto segmentado con el nuevo algoritmo (vistas coronal, axial y sagital). A. Atlas de fascículos de SWM (LNAO-SWM79), compuesto de 50 fascículos por hemisferio. B. Sujeto segmentado de 955K fibras. Fuente: elaboración propia.

El otro atlas se obtuvo utilizando la misma base de datos, pero aplicando una estrategia de agrupamiento de todo el cerebro y un registro no lineal (62 fascículos, 44.345 centroides) (Román et al. (2017)). Los centroides en ambos atlas tienen 21 puntos

equidistantes.

3.3.2. Segmentación de WM basada en atlas multisujeto

El método propuesto en Labra et al. (2017) clasifica las fibras de los sujetos de WM en función de un atlas de fascículos multisujeto (Guevara et al. (2012a)). Las fibras de los sujetos se clasifican usando un atlas de fascículos calculando la distancia máxima Euclidiana entre cada fibra del sujeto y cada centroide del atlas, y manteniendo solo las fibras con una distancia por debajo de un umbral definido para cada fascículo. La distancia máxima Euclidiana (d_{ME}) entre una fibra a y el centroide b , ambos de N puntos, se basa en la distancia Euclidiana d_E entre sus puntos 3D correspondientes a_i y b_i , donde:

$$d_E(a_i, b_i) = \|(a_i - b_i)\| \quad (3.1)$$

$$d_{ME}(a, b) = \min(\max_i(d_E(a_i, b_i)), \max_i(d_E(a_i, b_{N-i}))) \quad (3.2)$$

Debido a que la orientación espacial de las fibras en el conjunto de datos es desconocida, d_{ME} considera el mínimo de ambas orientaciones posibles (directa e inversa), como se muestra en la ecuación 3.2. La distancia de d_{ME} se normaliza luego agregando el término TN , que penaliza la diferencia entre las longitudes de la fibra del sujeto (l_S) y el centroide del atlas (l_C):

$$TN = \left(\frac{|l_S - l_C|}{\max(l_S, l_C)} + 1 \right)^2 - 1 \quad (3.3)$$

Antes del procesamiento, las fibras del sujeto se vuelven a muestrear utilizando 21 puntos equidistantes. El algoritmo clasifica las fibras del sujeto en dos pasos. Primero, la etapa de *preclasificación* descarta las fibras usando solo un subconjunto de puntos

3D para calcular d_{ME} , basado en el hecho que si $d_E(a_i, b_i)$ está por encima del umbral para al menos un valor de i , luego $d_{ME}(a, b)$ estará por encima del umbral, y la fibra a se descartará. A continuación, el paso de *clasificación* calcula la métrica de distancia completa ($d_{ME} + TM$) para las fibras que no se descartaron en la primera etapa.

La optimización del tiempo de ejecución del algoritmo se logra principalmente a través de:

- **Descarte de fibras progresivo:** para las fibras remuestreadas con 21 puntos, es necesario calcular 42 distancias $d_E(a_i, b_i)$, considerando direcciones directas e inversas. Para reducir el número de cálculos, el algoritmo descarta las fibras con la métrica de distancia simplificada utilizando primero solo el punto central (*test1*). Luego, las fibras supervivientes se descartan utilizando los puntos extremos (*test2*), y la tercera etapa (*test3*) descarta las fibras supervivientes utilizando 4 puntos intermedios. Finalmente, las fibras restantes se clasifican utilizando la métrica de distancia completa (*test4*) (ver Figura 3.2).
- **Ejecución paralela:** el conjunto de datos se divide en subconjuntos más pequeños para que quepan en la memoria de la computadora. Para cada subconjunto, las distancias entre las fibras del sujeto y los centroides del atlas se evalúan en paralelo utilizando múltiples hebras. Los subconjuntos se procesan secuencialmente.

3.3.3. Nuevo algoritmo optimizado

Se ha propuesto el algoritmo 1 para optimizar tanto la memoria como el tiempo de cálculo. La estrategia de utilizar las 4 etapas para descartar progresivamente las fibras se ha mantenido, pero se han incluido varias optimizaciones en el proceso de descarte. A continuación, se describen las optimizaciones más relevantes dentro del código:

- **Cantidad de memoria reservada:** la complejidad espacial del algoritmo original es $\mathcal{O}(NM)$, donde N es el número de fibras en el conjunto de datos del sujeto, y M es el número de centroides en el atlas. Para conjuntos grandes, este requisito

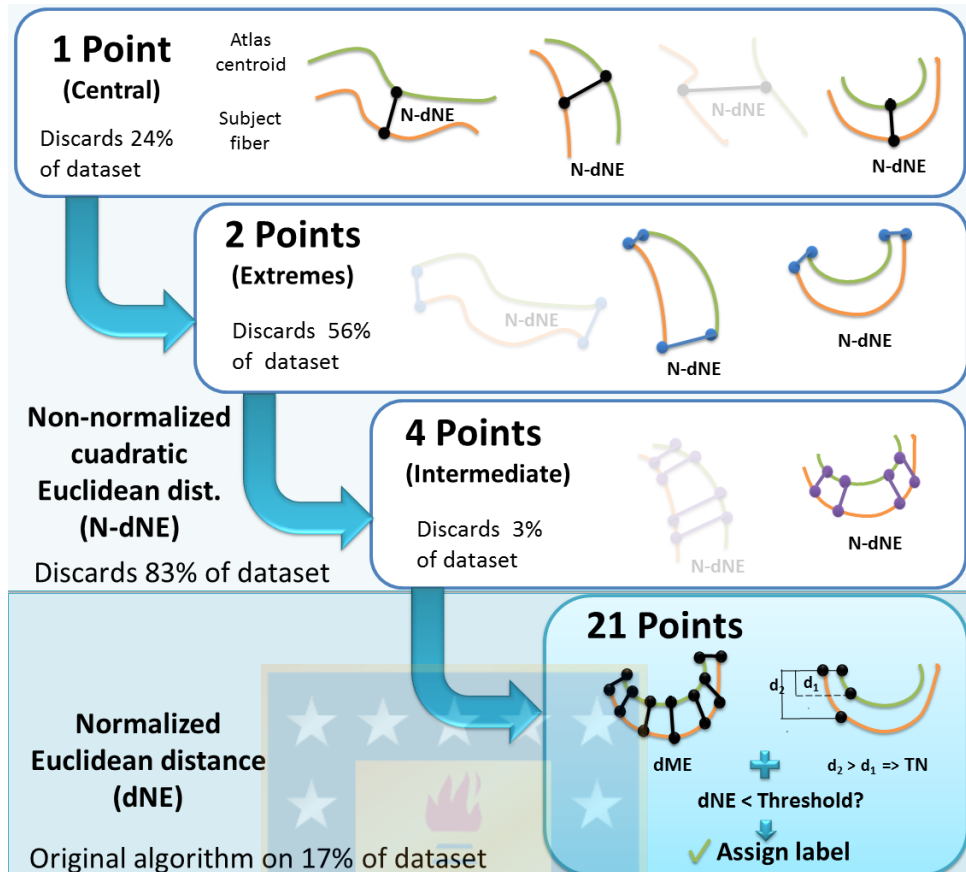


Figura 3.2: Descarte en 4 etapas del algoritmo de segmentación. Fase 1: descarte por el punto central. Fase 2: descarte utilizando los puntos de los extremos. Fase 3: descarte utilizando 4 puntos intermedios. Fase 4: clasificación de fibras mediante la métrica de la distancia Euclidiana normalizada. Fuente: Labra et al. (2017).

no es factible y los datos deben procesarse secuencialmente en fragmentos, lo que limita el paralelismo del algoritmo. En contraste, la complejidad espacial del nuevo algoritmo es solo $\mathcal{O}(N + M)$, porque compara una fibra del sujeto con cada centroide del atlas hasta que se le asigna una etiqueta. Esto nos permite cargar todo el conjunto de datos en la memoria y procesar todas las fibras en paralelo, incluso si la computadora tiene memoria limitada o recursos de archivos de intercambio.

- **Sincronización paralela:** intuitivamente, el algoritmo descarta las fibras en paralelo de forma independiente, por lo que no tiene problemas de colisión. Esto se logra utilizando un arreglo de tamaño igual al número de fibras del sujeto, donde

Algoritmo 1 Segmentación paralela

```

1: for i = 0 to nfibersSubject do
2:   for j = 0 to atlasData.size() do
3:     for k = 0 to (atlasData[j].size()/ndataFiber) do
4:       isInverted, isDiscarded ← false
5:       ed ← -1
6:       {test1}
7:       isDiscarded = discardCenter()
8:       if isDiscarded then continue end if
9:       {test2}
10:      isDiscarded = discardExtremes()
11:      if isDiscarded then continue end if
12:      {test3}
13:      isDiscarded = discardFourPoints()
14:      if isDiscarded then continue end if
15:      {test4}
16:      isDiscarded = discarded21points()
17:      if ed ≠ -1 then
18:        if ed < euclideanDistances[i] then
19:          euclideanDistances[i] ← ed {obtiene la mínima distancia Euclidiana}
20:          assignment[i] ← j {cada fibra se asigna al fascículo correspondiente}
21:        end if
22:      end if
23:    end for
24:  end for
25: end for
26: return assignment {devuelve el fascículo asociado con cada fibra}

```

se almacena el índice del fascículo al que pertenece cada fibra.

- **Localidad de los datos:** el algoritmo paralelo propuesto mejora la ubicación de los accesos a la memoria, haciendo así un uso más eficiente del sistema de memoria local de cada procesador. Esto es posible debido al anidamiento de los bucles, en el que cada procesador puede mantener las fibras candidatas para cada fascículo en su caché y RAM.
- **Inferencia de la dirección de la fibra:** de acuerdo con la ecuación 3.2, el algoritmo original calcula la distancia entre las fibras y los centroides en ambas direcciones en cada etapa, lo que conlleva a operaciones duplicadas. Nuestro

algoritmo utiliza las distancias calculadas en la segunda etapa de descarte para determinar la dirección de la fibra y almacenarla. Por lo tanto, la tercera y cuarta etapas se evalúan en una sola dirección, reduciendo a la mitad la cantidad de cómputo y eliminando la operación $\min()$ de la ecuación 3.2.

Otra mejora es que el algoritmo evalúa el criterio de descarte después de cada cálculo de distancia por pares $d_E(a_i, b_i)$. Si la distancia está por encima del umbral para el fascículo correspondiente, la evaluación se detiene y la fibra se descarta inmediatamente.

Finalmente, la etapa de clasificación final de la versión anterior agrega el término de normalización TN a todas las distancias antes de compararlas con el umbral. En nuestro nuevo algoritmo, si d_{ME} es mayor que el umbral, descarta la fibra inmediatamente sin calcular TN .

- **Clasificación de las fibras:** el algoritmo original, probado con fascículos de materia blanca profunda (DWM), puede asignar una fibra a varios fascículos. Esto no es un problema con el atlas de DWM (Guevara et al. (2012a)), porque rara vez etiqueta una fibra con dos fascículos. Con su implementación paralela original, no fue posible actualizar eficientemente la distancia mínima para varios fascículos y mantener el mínimo. Para los fascículos de SWM, es muy común satisfacer el criterio de umbral para dos fascículos vecinos, por lo que modificamos el algoritmo para mejorar la clasificación y realizar el etiquetado solo para el fascículo más cercano.

3.4. MÉTODO DE CLUSTERING DE FIBRAS

En esta sección se describen algunos conceptos generales para dar una mayor comprensión al método de clustering de fibras desarrollado. Entre ellos destacan el algoritmo de Mini Batch K-means, el método Elbow, Clique Maximal y el cálculo de los centroides del método. Finalmente se detallará el método de clustering implementado haciendo uso de esas técnicas.

3.4.1. Mini Batch K-means en clustering de puntos de fibras

Mini Batch K-means es un algoritmo para el clustering masivo de datos (Sculley (2010)). El objetivo principal es representar el conjunto completo de datos por un subconjunto del mismo. El pseudocódigo se muestra en el algoritmo 2. Este algoritmo se basa en la idea de utilizar pequeños lotes de muestras tomados al azar con un tamaño fijo para que puedan almacenarse en la memoria principal. Para cada iteración del bucle y hasta que el algoritmo converja, se toman pequeños lotes elegidos al azar del conjunto de datos y se utilizan para actualizar los clústeres, asignando un grupo a cada punto de datos del lote, teniendo en cuenta la ubicación anterior de los centroides de los clústeres. Luego, cada mini batch debe actualizar los clústeres utilizando una combinación de los valores de los ejemplos y los prototipos, teniendo en cuenta las ubicaciones anteriores de los centroides del clúster, mientras aplica una velocidad de aprendizaje que disminuye a medida que avanza el número de iteraciones.

El número de ejemplos que se asignan a un clúster en el proceso es el inverso de la velocidad de aprendizaje. A medida que crece el número de iteraciones, los nuevos ejemplos pierden efecto y el algoritmo converge cuando no hay cambios en los clústeres al pasar varias iteraciones. La gran ventaja de este algoritmo es que realiza un submuestreo de los datos en cada iteración, en contraste con lo que hace el algoritmo K-means (Steinbach et al. (2000)), reduciendo así el costo computacional.

Algoritmo 2 Mini Batch K-means

```

1: k clústeres, mini-batch mb, data set D, número de iteraciones it
   Para cada c inicializado  $\in C$ , x se toma aleatoriamente desde X
2:  $v \leftarrow 0$ 
3: for i = 0 to it do
4:    $M \leftarrow mb$  {Las muestras tomadas aleatoriamente de X}
5:   for x  $\in M$  do
6:      $d[x] \leftarrow f(C, x)$  {El centro más cercano a x debe almacenarse en la caché}
7:   end for
8:   for x  $\in M$  do
9:      $c \leftarrow d[x]$  {En este paso tienes que obtener el centro en la caché para este x}
10:     $v[c] \leftarrow v[c] + 1$  {Actualiza los conteos que existen por centro}
11:     $\eta \leftarrow 1/v[c]$  {Obtiene la tasa de aprendizaje por centro}
12:     $c \leftarrow (1 - \eta)c + \eta x$  {En este paso se realiza el gradiente}
13:   end for
14: end for

```

3.4.2. Método Elbow

El método Elbow se utiliza para encontrar el número óptimo de clústeres en un conjunto de datos determinado. La idea básica es elegir la cantidad de clústeres para que, si se agrega otro clúster, no produzca un cambio significativo de los datos. El algoritmo 3 muestra el pseudocódigo del método Elbow.

Al aplicar K-means a diferentes números de clústeres (de 1 a N), el método Elbow utiliza los valores obtenidos de la inercia (ver eq. 3.4), siendo la inercia la suma de las distancias al cuadrado de cada elemento del clúster al centroide:

$$Inertia = \sum_{i=0}^n \|x_i - \mu\|^2 \quad (3.4)$$

Cuando los valores de la inercia se obtienen después de aplicar el algoritmos de K-means de 1 a N clústeres, la inercia se representa de manera que el porcentaje de variación de los clústeres se realice de acuerdo con el número de clústeres. En la gráfica se debe apreciar un cambio en la trayectoria de la inercia, por lo que debe ser

Algoritmo 3 Método Elbow para determinar el k óptimo

```
1:  $distortions[] \leftarrow 0$   
2: for  $k = 0$  to  $maxk$  do  
3:    $kmeans \leftarrow KMeans(k, randomState = 0)$  {Inicializa los centroides aleatoria-  
   mente}  
4:    $kmeans.fit(X)$  {Cálculo del clustering K-means}  
5:    $distortions \leftarrow kmeans.inertia$  {Calcular inercia}  
6: end for
```

similar a la forma de un brazo y su codo. El punto exacto donde se puede ver que el cambio en la inercia es el que proporciona el número óptimo de clústeres a seleccionar en el conjunto de datos, es decir, el punto que muestra el codo del brazo es el número óptimo de clústeres para el conjunto de datos, como se puede ver en la figura 3.3.

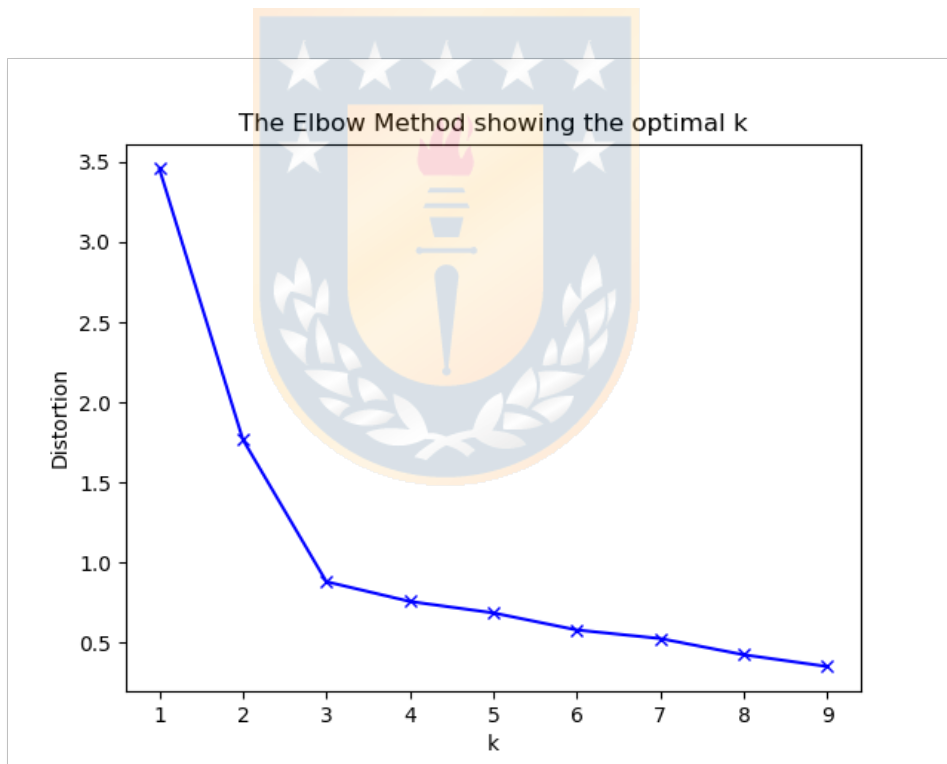


Figura 3.3: Método Elbow. El número óptimo de clústeres viene dado por el cambio en la inercia, es decir, el punto que muestra el codo del brazo ($k=3$). Fuente: <https://pythonprogramminglanguage.com/kmeans-elbow-method/>.

3.4.3. Clique y clique maximal

Un clique es un problema de decisión que pertenece a los 21 problemas NP-completos de Karp (Karp (1972)). Por lo tanto, encontrar un clique C en un grafo no dirigido $G = (V, E)$, donde V es el número de vértices y E es el número de aristas, es equivalente a encontrar un subconjunto de vértices W , todos adyacentes entre sí formando los subgrafos completos $G(C)$. El tamaño, por lo tanto, será el número de vértices que contiene el clique. Una formulación de este problema es encontrar el clique maximal, que es un problema de optimización, por lo que se vuelve NP-hard, en el que ya no se pueden extender más vértices en un grafo dado. Debido a que puede haber una cantidad exponencial de cliques maximales en un grafo, encontrar todos los cliques maximales tiene una complejidad de tiempo exponencial en el peor caso (Sipser (2006)).

El grafo formado por 6 nodos en la Figura 3.4 muestra dos cliques maximales C_1 y C_2 en un grafo G , donde $C_1 = \{1, 3, 4, 5\}$ y $C_2 = \{1, 2, 6\}$. Ambos son cliques maximales porque son cliques, y específicamente en C_1 los nodos 2 y 6 no tienen una arista a los demás nodos del subgrafo, y en C_2 ninguno de los vértices de C_1 tiene una arista que los enlace con los nodos de C_2 .

3.4.4. Cálculo de los centroides

Se ha implementado un nuevo algoritmo para el cálculo de centroides de los clústeres. A pesar de que la biblioteca Dipy (Garyfallidis et al. (2014)) ya contiene esta funcionalidad, su cálculo de centroides presenta ciertos problemas en nuestro algoritmo de clustering.

Dipy realiza el cálculo de centroides de manera muy simple: calculando la media aritmética de cada uno de los puntos de las fibras. Este método funciona bien para la gran mayoría de sus aplicaciones, sin embargo, en nuestro código de clustering devuelve centroides erróneos. Esto es debido a que, después de la etapa de reasignación de clústeres, en los que se unen clústeres pequeños a los clústeres más grandes,

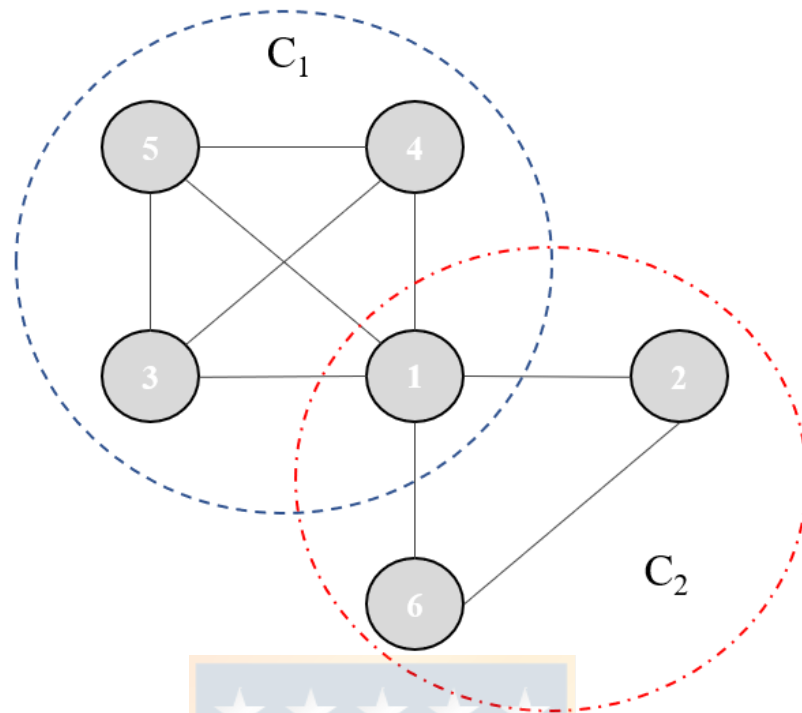


Figura 3.4: Clique maximal. El grafo G tiene dos cliques maximales, C_1 formado por los vértices $\{1, 3, 4, 5\}$ y C_2 compuesto por los vértices $\{1, 2, 6\}$. Fuente: elaboración propia.

las fibras no mantienen un orden, es decir, pueden estar invertidas unas con respecto a las otras y, por eso, al calcular la media aritmética sobre los puntos de las fibras desordenadas, se obtienen centroides no válidos. Otro problema del método, es que, al calcular la media, se obtiene una fibra nueva, que realmente no está presente en la tractografía, y que puede tener una forma muy diferente a las fibras del clúster.

A continuación, se describe el método implementado para realizar el cálculo de centroides de los clusters:

1. Dado un clúster, se seleccionan del 60 % al 80 % de las fibras más largas. El motivo es que las fibras demasiado pequeñas normalmente no son representativas en un clúster.
2. De las fibras seleccionadas en el paso anterior, se seleccionan un 10 % aleatorio. Esta selección, es para que, en los siguientes cálculos, el tiempo de cómputo sea

muy bajo.

3. Se calcula la matriz de distancias de las fibras anteriores consigo mismas y se selecciona como centroide, la fibra que está más cerca de las demás. De esta manera, se obtiene como centroide, una fibra que está presente en el clúster y además el cálculo es muy rápido, ya que se seleccionan muy pocas fibras para realizar la matriz de distancias.

3.4.5. Nuevo método: CENTELLA

El método propuesto en este trabajo es un algoritmo de clustering de particionamiento rápido para las fibras de la materia blanca. Se basa en el hecho de que los clústeres buscados contendrán fibras similares en toda su extensión, es decir, sus puntos correspondientes estarán cerca en el espacio 3D. Por lo tanto, el clustering de algunos puntos de las fibras se aplica por separado para cada punto, lo que permite también un mayor rendimiento. Los clústeres de puntos se utilizan para agrupar fibras similares. Finalmente, se realiza otro postprocesamiento para disminuir el número final de clústeres.

El esquema propuesto toma como entrada un conjunto de datos de tractografía cerebral de un sujeto individual. Las fibras se vuelven a muestrear en un conjunto fijo de 21 puntos equidistantes y tienen formas 3D en cualquier orientación en el espacio, lo que significa que se pueden almacenar en la memoria en orden directo o inverso.

Sea T_s un conjunto de datos de tractografía de un sujeto individual que consiste en una colección de fibras, donde cada fibra está formada por 21 puntos en el espacio 3D.

Debido a que las fibras se pueden almacenar en sentido directo e inverso, se denota una fibra en sentido directo como a y su representación invertida como a^F . Se asume una distancia Euclidiana directa (d_E) e inversa (d_{EF}), y se considera la máxima distancia Euclidiana (d_{ME}) para medir la distancia entre las fibras a y b , como se puede ver definida en la Eq.3.5.

$$\begin{aligned}
d_E(a, b) &= \|a - b\| = \left(\sum_{i=0}^{20} (a_i - b_i)^2 \right)^{1/2} \\
d_{EF} &= d_E(a, b^F) = d_E(a^F, b) \\
d_{ME} &= \min(\max(d_E(a, b)), \max(d_{EF}(a, b)))
\end{aligned} \tag{3.5}$$

El método CENTELLA consiste en los siguientes cuatro pasos: (1) construcción de los clústeres de puntos, (2) generación de clústeres de fibras preliminares, (3) reasignación de clústeres pequeños de fibras preliminares y (4) fusión de clústeres de fibras candidatos (ver Figura 3.5). Este método es una mejora del método propuesto por Sanchez et al. (2018).

■ PASO 1: Construcción de los clústeres de puntos

El objetivo de este paso es generar clústeres de puntos. Para ello se utiliza el algoritmo de Mini-Batch K-means (Sculley (2010)) en un subconjunto de puntos de la fibra, porque de este modo es más rápido. Permite al usuario seleccionar el número de clústeres. En este paso el método recibe como entrada el número de clústeres y 5 puntos de cada fibra.

Las fibras contienen 21 puntos equidistantes y de ellos se eligen 5 puntos. Al aplicar el algoritmo de clustering en cada punto de las fibras de manera independiente, el número de clústeres no necesita ser el mismo en todos los puntos de las fibras. De hecho, se usan diferentes números para los puntos extremos de la fibra y el punto central. Se denota el número de clústeres para los puntos extremos como K_{p_o} , y el número de clústeres para el punto central como K_{p_c} . Se estimó el número de clústeres en cada punto de la fibra utilizando el método Elbow (Kodinariya and Makwana (2013)). Dado que aplicar Mini-batch K-means

en puntos de la fibra es una tarea independiente, se realiza este paso en paralelo. Por ejemplo, la Figura 3.5-1 muestra este paso aplicando Mini-batch K-means en cinco puntos de la fibra (puntos 0, 3, 10, 17, 20).

La razón por la que se toman 5 puntos en vez de 21 es obtener un mejor tiempo de ejecución del algoritmo. Además, que sean 5 puntos equidistantes determinados permite que los clústeres de puntos obtenidos sean uniformes tanto en los extremos como en los puntos centrales. Nótese que los puntos de los extremos se toman para que las fibras no presenten extremos muy dispersos. Con esos 5 puntos es suficiente ya que se ha mostrado en las pruebas que realiza una buena clasificación.

Después de aplicar el algoritmo de clustering, hay clústeres de puntos de fibras identificados por etiquetas, donde los clústeres en el primer punto de la fibra se identifican por las etiquetas A y F , en el cuarto punto de la fibra son B y G , en el undécimo punto es C , en el decimoctavo punto son D y H , y en el vigésimo primero punto son E e I . Finalmente, la salida de este paso son clústeres de puntos.

En resumen, este paso realiza Mini-Batch K-means en paralelo en los puntos de la fibra. Los parámetros de este paso son el número de puntos de la fibra a considerar para el clustering y el número de clústeres a utilizar en Mini-Batch K-means para cada punto de la fibra. Se usan K_{p_c} clústeres en puntos centrales y K_{p_o} clústeres en los extremos de la fibra. Al final, se crean clústeres de puntos.

■ PASO 2: Generación de clústeres de fibras preliminares

En el paso anterior, se han obtenido clústeres de puntos, agrupando por ejemplo todos los puntos 0 o todos los puntos 20. El objetivo de este paso es mapear esos clústeres de puntos a clústeres de fibras ya que finalmente lo que devuelve el algoritmo son clústeres de fibras preliminares. Este paso no necesita ningún

parámetro.

Cada fibra se identifica con la etiqueta que devuelven sus clústeres en el paso anterior. Entonces, se generan clústeres preliminares agrupando fibras que comparten la misma etiqueta que los clústeres de puntos de fibras. Esto se ha logrado mediante la implementación de un diccionario donde cada clave es dada por un conjunto de etiquetas de clúster, a los que pertenecen los puntos de la fibra, y fibras que comparten el mismo conjunto de etiquetas de clúster conformando un clúster preliminar.

Por ejemplo, la Figura 3.5-2 muestra dos clústeres preliminares identificados por las etiquetas de puntos de la fibra (A, B, C, D, E) y (F, G, C, H, I) , donde el primer clúster preliminar consiste en las fibras (p, q) y el segundo está formado por las fibras (r, s) . Como salida de este paso se devuelven clústeres de fibras preliminares.

■ PASO 3: Reasignación de clústeres pequeños de fibras preliminares

Este paso requiere dos parámetros. Uno es el *minsize*, número de fibras en clústeres para dividir los clústeres preliminares en los conjuntos S_L (clústeres grandes) y S_S (clústeres pequeños), y el d_{Rmax} que es la distancia máxima Euclidiana para realizar la reasignación.

Durante este paso se realiza una reasignación de clústeres preliminares. El objetivo es reducir el número de clústeres pequeños que existen al reasignarlos a clústeres más grandes. Dado que el paso anterior genera muchos clústeres formados por solo unas pocas fibras, se reasignan los clústeres pequeños a los clústeres más grandes en función de algunas condiciones. Se consideran tanto el tamaño como un umbral de la distancia máxima Euclidiana (directa e inversa) para decidir si se deben reasignar o no los clústeres preliminares.

Para realizar esto, primero se dividen los clústeres preliminares en dos conjuntos basados en el número de fibras que contienen. Un conjunto S_L contiene todos los clústeres preliminares con un número de fibras sobre un *minsize* y el conjunto S_S contiene todos los clústeres preliminares con número de fibras por debajo o igual de *minsize*. Aquellos que contienen 5 fibras o menos son clústeres pequeños (S_S) y los otros son clústeres grandes (S_L).

A continuación, se calculan los centroides para cada clúster preliminar en ambos conjuntos. Entonces, un clúster preliminar en el conjunto S_S es reasignado al clúster preliminar más cercano en el conjunto S_L , solo si la distancia entre sus centroides está por debajo de un umbral $d_E(a, b) < d_{Rmax}$ (donde $d_E(a, b)$ está definido como en la Eq. 3.5). Si la distancia Euclídea entre centroides no es menor que el umbral entonces los clústeres no se reasignan.

Al final de este paso sucede que si hay clústeres preliminares en el conjunto S_S y contienen una fibra, son considerados ruido y se eliminan. La Figura 3.5-3 muestra a los clústeres preliminares separados en conjuntos (Figura 3.5-3.1.), cálculo de centroides (Figura 3.5-3.2.), casos de reasignación, sin reasignación y eliminación (Figura 3.5-3.3.). Al final se obtienen *clústeres candidatos*. Nótese que durante la reasignación, se conservan las etiquetas de los clústeres obtenidos del paso previo.

El código de segmentación presentado en el apartado 3.3.3 se adaptó para esta fase del algoritmo. El atlas de referencia para realizar la comparación y clasificación de las fibras pasó a ser el conjunto de clústeres grandes S_L . Por otra parte, las fibras del sujeto a clasificar, en este trabajo son los clústeres pequeños S_S .

Para resumir, este paso recibe como entrada el número de fibras de los clústeres y el umbral de distancia. Los clústeres se dividen en grandes y pequeños y la reasignación se realiza cuando la distancia Euclídea entre los centroides de los clústeres es menor que el umbral. Si la distancia es mayor, los clústeres

no se reasignan. Además, si no se reasigna y el clúster tiene una sola fibra, se considera ruido y se elimina. Finalmente se obtienen los clústeres candidatos.

■ PASO 4: Fusión de clústeres de fibras candidatos

El último paso recibe como entrada el umbral de distancia y los clústeres obtenidos del paso previo.

El objetivo es fusionar clústeres candidatos cercanos que comparten la etiqueta del punto central previamente obtenida del paso 1, y reducir el número de clústeres haciéndolos más poblados de fibras. Se escogió el punto central debido a que los clústeres de fibras están cercanos en su punto central y en los extremos están usualmente más separados. Se fusionan los clústeres cuyos centroides presentan distancias entre ellos por debajo del umbral de distancia máxima Euclidiana (directa o inversa).

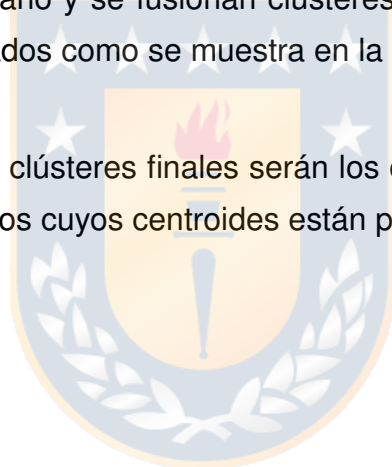
Para evitar el cómputo de todas las posibles configuraciones de los múltiples clústeres candidatos que solo pueden cumplir con esta restricción, se formula el problema utilizando una representación de grafo y se aproxima la solución utilizando un algoritmo de grafos. Se define un grafo no dirigido como $G = (V, E)$, donde un vértice $v \in V$ representa un centroide como un clúster candidato, y existe una arista $e = (u, v) \in E$ en G solo si $d_{ME}(u, v) < d_{Mmax}$. En otras palabras, hay una arista entre dos vértices, solo si dos centroides están por debajo de una distancia máxima Euclidiana dada por d_{Mmax} . Nótese que la definición de d_{ME} está en la Eq 3.5.

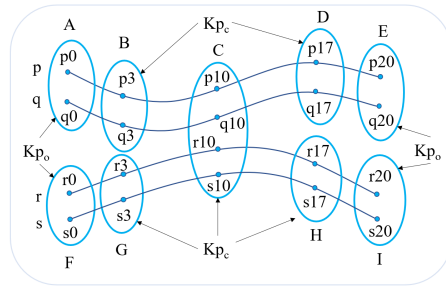
Luego, el objetivo es encontrar subgrafos densos en G donde todos los vértices en el subgrafo están conectados entre ellos. Este patrón de subgrafo denso captura la idea de que todos los clústeres candidatos que están cerca uno del otro se deben fusionar en uno solo. En la teoría de grafos, este patrón se conoce como

clique (Karp (1972)). Más específicamente, el objetivo es encontrar *cliques maximales* en G , es decir, cliques que no se pueden ampliar agregando un vértice adyacente más. En otras palabras, un *clique maximal* no es un subconjunto de un *clique* más grande.

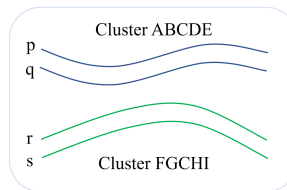
Encontrar cliques maximales es un problema NP-hard (Woeginger (2003); Sipser (2006)), pero la complejidad temporal es menor en grafos esparsos (Eppstein and Strash (2011)). En nuestra representación esperamos no tener un grafo G ni grande ni denso. De hecho, solo se consideran los centroides de los clústeres candidatos como vértices y el número de aristas en G se espera que sea pequeño dado que solo una pequeña fracción de todos los posibles pares de centroides satisfacen d_{Mmax} . Después de encontrar cliques maximales en G mayores a dos, se ordenan por tamaño y se fusionan clústeres candidatos solo si no han sido previamente fusionados como se muestra en la Figura 3.5-4.

Como resultado, los clústeres finales serán los clústeres candidatos o clústeres candidatos fusionados cuyos centroides están por debajo de la distancia d_{Mmax} .

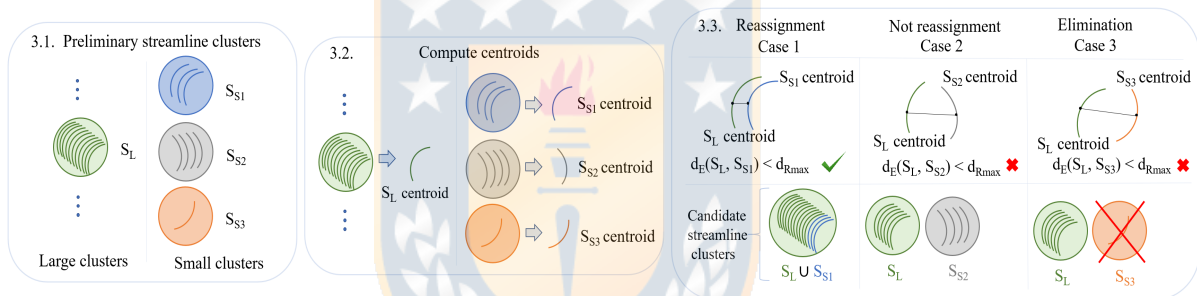




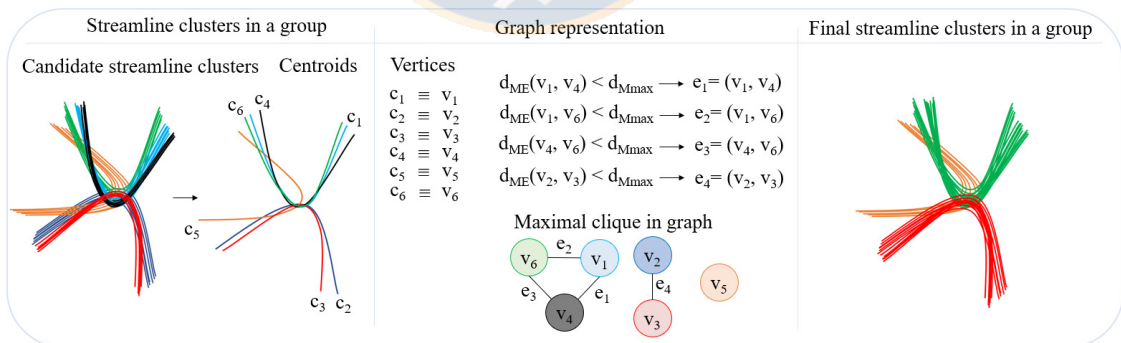
PASO 1: Construcción de los clústeres de puntos.



PASO 2: Generación de clústeres de fibras preliminares.



PASO 3: Reasignación de clústeres pequeños de fibras preliminares.



PASO 4: Fusión de clústeres de fibras candidatos.

Figura 3.5: CENTELLA. PASO 1: Construcción de los clústeres de puntos. Mini-Batch K-means se aplica en paralelo en los puntos marcados, generando clústeres que posteriormente se marcan con la misma etiqueta. PASO 2: Generación de clústeres de fibras preliminares. Todas las fibras que comparten la misma etiqueta formarán el mismo clúster. PASO 3: Reasignación de clústeres pequeños a clústeres más grandes con fibras preliminares. Los clústeres se separan en conjuntos grandes y pequeños (3.1.). Se calculan los centroides (3.2.). Finalmente ocurren 3 casos: se reasignan, no se reasignan o se eliminan (3.3.). PASO 4: Fusión de clústeres de fibras candidatos. El objetivo es reducir el número de clústeres mediante cliques maximales. Fuente: elaboración propia.

Capítulo 4

RESULTADOS

4.1. INTRODUCCIÓN

En el presente capítulo se describen las pruebas realizadas tanto para el método de segmentación como para el método de clustering de fibras.

4.2. RESULTADOS SEGMENTACIÓN DE FIBRAS

El algoritmo optimizado se implementó en C++ 11, compilado con gcc 7.3.0. Hemos realizado nuestros experimentos en una computadora con una CPU Intel Core i7-6700K de 8 núcleos a 4 GHz, 8 MB de caché L3 compartida y 8 GB de RAM. El sistema operativo es Ubuntu 18.04.1 LTS con kernel 4.15.0-36 (64 bits).

Para evaluar el rendimiento de los algoritmos, utilizamos un conjunto de datos de tractografía determinística de un sujeto. La versión optimizada del algoritmo produce casi los mismos resultados que el algoritmo anterior, excepto que clasifica las fibras en solo el fascículo más cercano, por debajo del umbral de distancia.

Los experimentos utilizan conjuntos de datos de sujetos remuestreados de 600.000 a aproximadamente 5.216.000 de fibras. La Figura 4.1 muestra el tiempo de ejecución de ambos algoritmos para diferentes tamaños de conjuntos de datos de sujetos, utilizando el atlas de 62 fascículos (Román et al. (2017)). Para un conjunto de datos de 1.634.000 fibras, nuestro nuevo algoritmo paralelo se ejecuta 2,15 veces más rápido que su versión anterior. Con un conjunto de datos de 4.145.000 fibras, el algoritmo es 2,34 veces más rápido que su predecesor. El algoritmo original segmenta un conjunto de datos de un sujeto de 5.216.000 fibras en aproximadamente 17 minutos, mientras

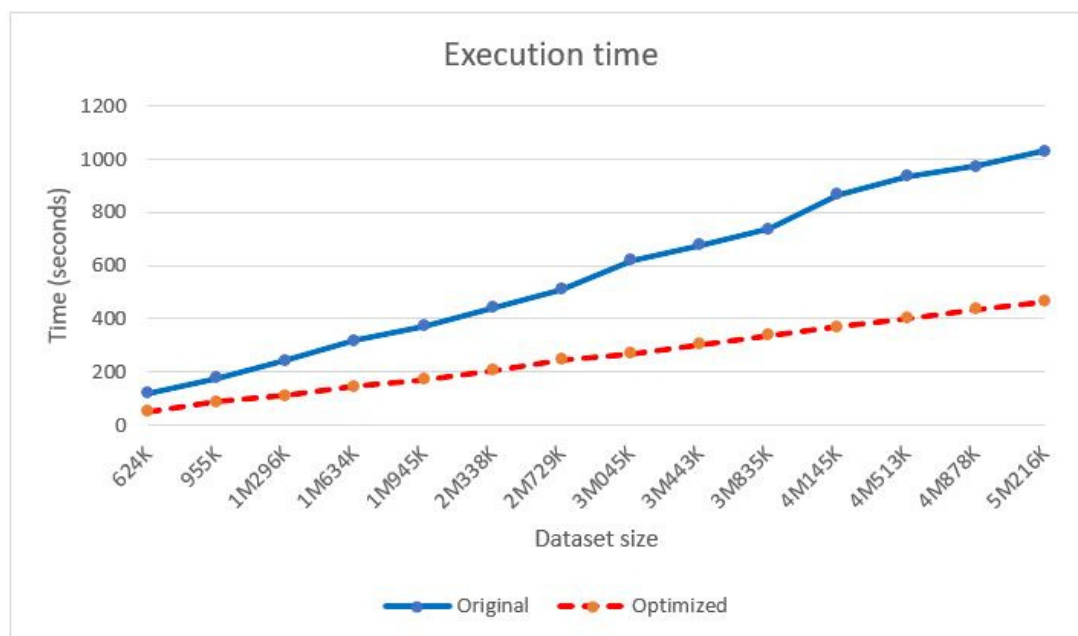


Figura 4.1: Tiempo de ejecución para ambos algoritmos en función del tamaño del conjunto de datos de fibras del sujeto. Fuente: elaboración propia.

que nuestro algoritmo segmenta el mismo conjunto de datos en menos de 8 minutos.

La Figura 4.2 muestra la memoria utilizada por ambos algoritmos utilizando el atlas compuesto por 100 fascículos de Guevara et al. (2017). Para el primer algoritmo, el uso de la memoria refleja la división del conjunto de fibras del sujeto en fragmentos que realiza el algoritmo para ajustar los datos en la memoria. En nuestro nuevo algoritmo, el gráfico muestra la memoria reservada por el algoritmo para todo el conjunto de datos. Incluso con el conjunto de datos completo en la memoria, el nuevo algoritmo consume constantemente menos memoria que el original. Como ejemplo, para el conjunto de datos de 3.443.000 de fibras, el algoritmo anterior consume 1,27 veces más RAM que el nuevo. El uso de la memoria crece linealmente para ambos algoritmos porque el tamaño del atlas es fijo.

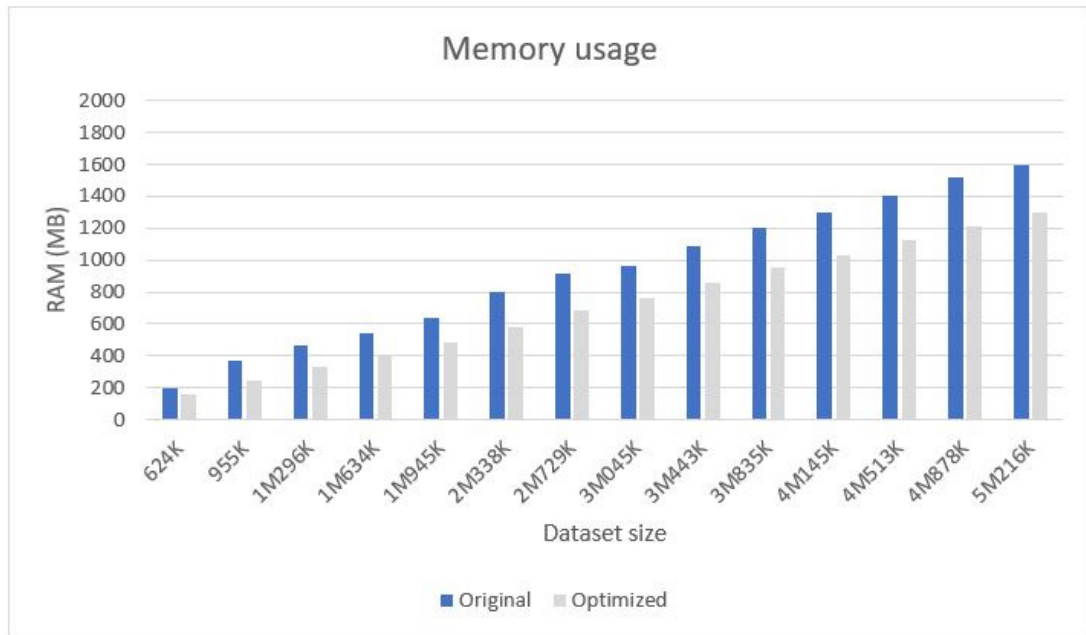


Figura 4.2: Uso de memoria para ambos algoritmos en función del tamaño del conjunto de datos de fibras del sujeto. Fuente: elaboración propia.

4.3. RESULTADOS PARA EL CLUSTERING DE FIBRAS

Para el método de clustering de fibras se han realizado múltiples pruebas. Se ha buscado la configuración óptima de parámetros, se han realizado pruebas cualitativas y cuantitativas para medir la calidad de los clústeres y finalmente se ha comparado el método con el mejor método de clustering exploratorio de fibras presente en el estado del arte, QuickBundles (Garyfallidis et al. (2012)), tanto en calidad como en tiempo de ejecución.

4.3.1. Configuración de parámetros para el análisis cuantitativo

El algoritmo presenta tres parámetros configurables: el número de clústeres para cada uno de los cinco puntos sobre los que se aplica el algoritmo K-means, el umbral de distancia máxima Euclidiana (d_{Rmax}) para la etapa de reasignación de clústeres pequeños a clústeres grandes y el umbral de distancia máxima Euclidiana (d_{Mmax}) para realizar la unión de clústeres en la última etapa.

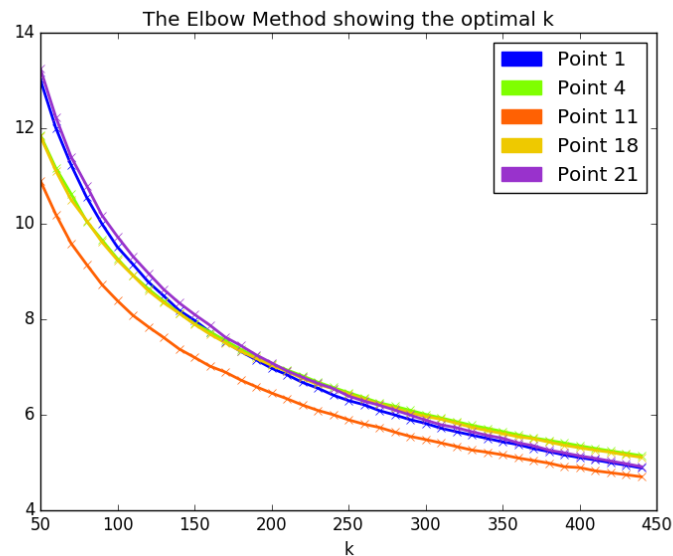


Figura 4.3: Método Elbow mostrando el k óptimo. En el eje x se muestra el número de clústeres, en el eje y se muestra la inercia. Los valores óptimos de k se sitúan en el “codo” de la línea. Fuente: elaboración propia.

- **Encontrar el número óptimo de clústeres (buscando el k óptimo).**

Se han realizado numerosas pruebas para determinar el número óptimo de clústeres (k) en cada uno de los cinco puntos durante la etapa de clustering con K-means (Figura 3.5-1 PASO 1). En primer lugar, se aplicó el método Elbow para obtener el número óptimo de clústeres en cada punto, para ello, se ejecutó el algoritmo K-means en cada uno de los cinco puntos de un sujeto de un millón de fibras de la base de datos ARCHI con 50, 150, 200, 250, 300, 350, 400 y 450 clústeres en cada ejecución (ver Figura 4.3).

Como se observa en la Figura 4.3, el número óptimo de clústeres en los puntos centrales ronda entre 150 y 200 clústeres y en los puntos externos entre 150 y 300. Sin embargo, se puede apreciar que los “codos” no están muy pronunciados y, por ese motivo, se han realizado más pruebas.

Adicionalmente a la aplicación del método Elbow, se realiza un análisis cuantitativo en función del parámetro k y diferentes criterios de calidad, que son la distancia máxima intraclúster y el tamaño de los clústeres. De esta manera, se

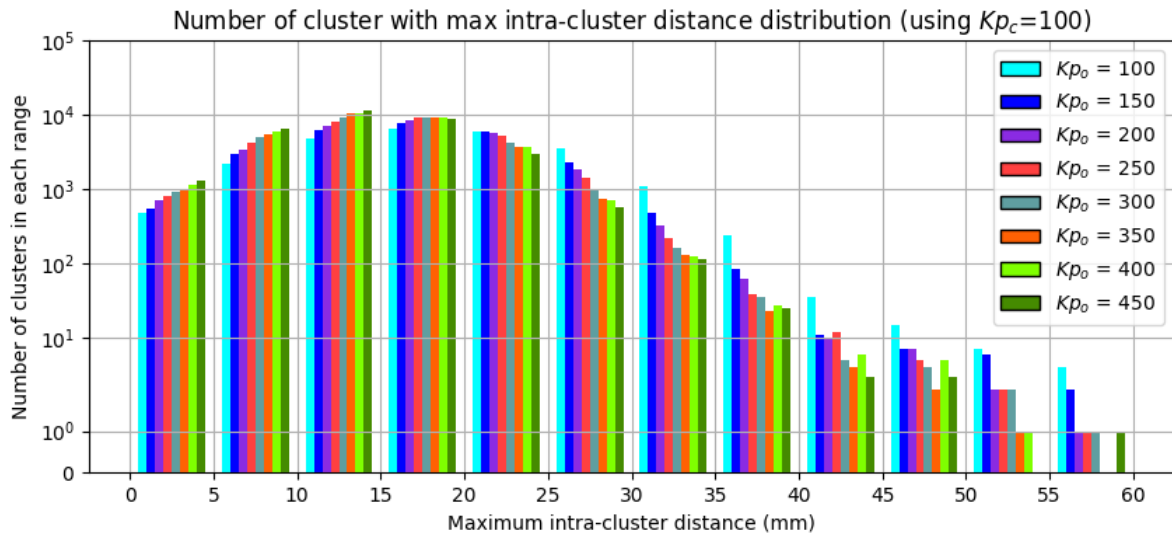


Figura 4.4: Histograma de distancia máxima intraclúster con 100 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función de la distancia máxima intraclúster utilizando $Kp_c = 100$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 100, 150, 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.

determina la calidad de los clústeres en función de k . Dicho análisis se realiza mediante la creación de múltiples histogramas.

En consecuencia, se han realizado tres histogramas de distancia máxima intraclúster, utilizando como número de clústeres en los puntos centrales (Kp_c), 100, 150 y 200 clústeres, y, para cada uno de esos valores, se han utilizado 150, 200, 250, 300, 350, 400 y 450 clústeres en los puntos extremos (Kp_o). Como medida de calidad se ha utilizado la distancia máxima intraclúster, que es la distancia máxima Euclidiana entre cada par de fibras de cada clúster. Los clústeres de calidad deben ser homogéneos y con distancias intraclúster no demasiado grandes (ver Figuras 4.4, 4.5 y 4.6).

Como se puede observar en las Figuras 4.4, 4.5 y 4.6, los resultados con los diferentes valores son muy similares entre ellos, es decir, la distancia máxima intraclúster resultante de la variación de los parámetros Kp_o y Kp_c cambia muy

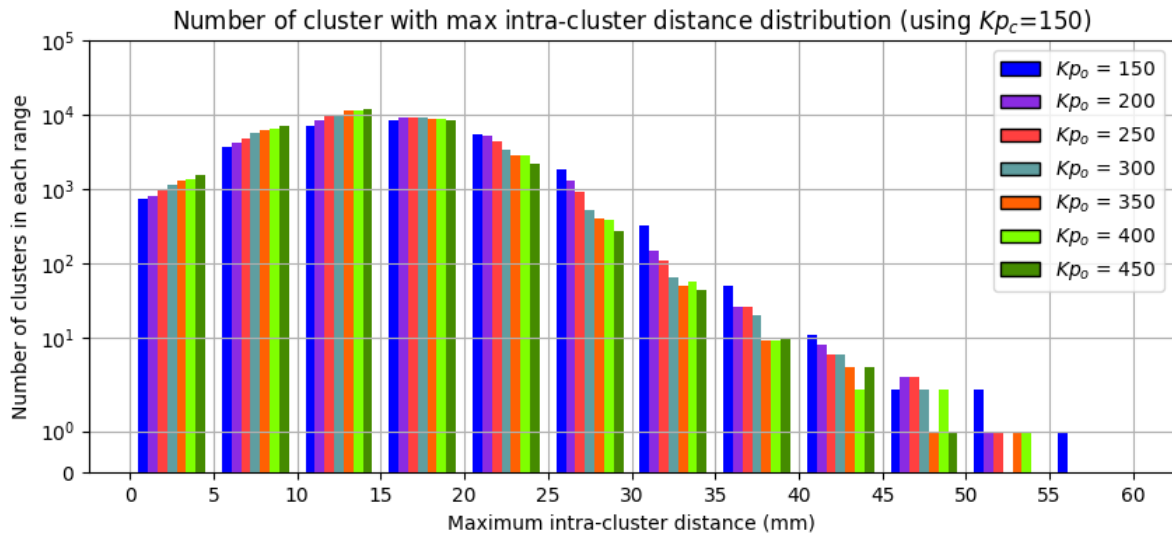


Figura 4.5: Histograma de distancia máxima intraclúster con 150 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función de la distancia máxima intraclúster utilizando $Kp_c = 150$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 150, 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.

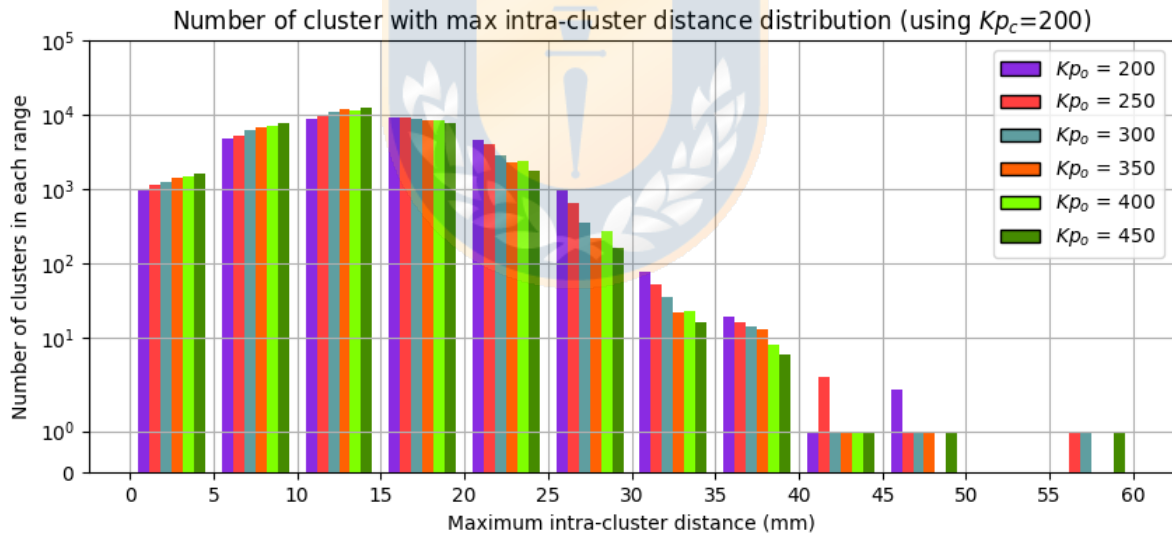


Figura 4.6: Histograma de distancia máxima intraclúster con clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función de la distancia máxima intraclúster utilizando $Kp_c = 200$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.

poco en las diferentes ejecuciones y, por lo tanto, los resultados finales son muy similares independientemente de los valores de k que se utilicen.

Por otro lado, es importante tener en cuenta que los clústeres realmente presentan valores de distancias intraclúster desde 1mm a 40mm, ya que se ha verificado visualmente que todos los resultados con distancia entre 40mm y 60mm no son clústeres normales, sino ruido en la tractografía. Esto se muestra más en detalle en el apartado 4.3.3.

Finalmente, se han realizado más pruebas con diferentes valores de Kp_c y Kp_o , pero utilizando como medida de calidad el tamaño de los clústeres para observar la variación del número de clústeres en función de los parámetros Kp_c y Kp_o . Dichas pruebas se han realizado con la misma configuración de parámetros que las pruebas de distancia intraclúster. Se han utilizado $Kp_o = 100, 150$ y 200 y $Kp_c = 100, 150, 200, 250, 300$ y 350 , dando como resultado tres histogramas (ver Figuras 4.7, 4.8 y 4.9).

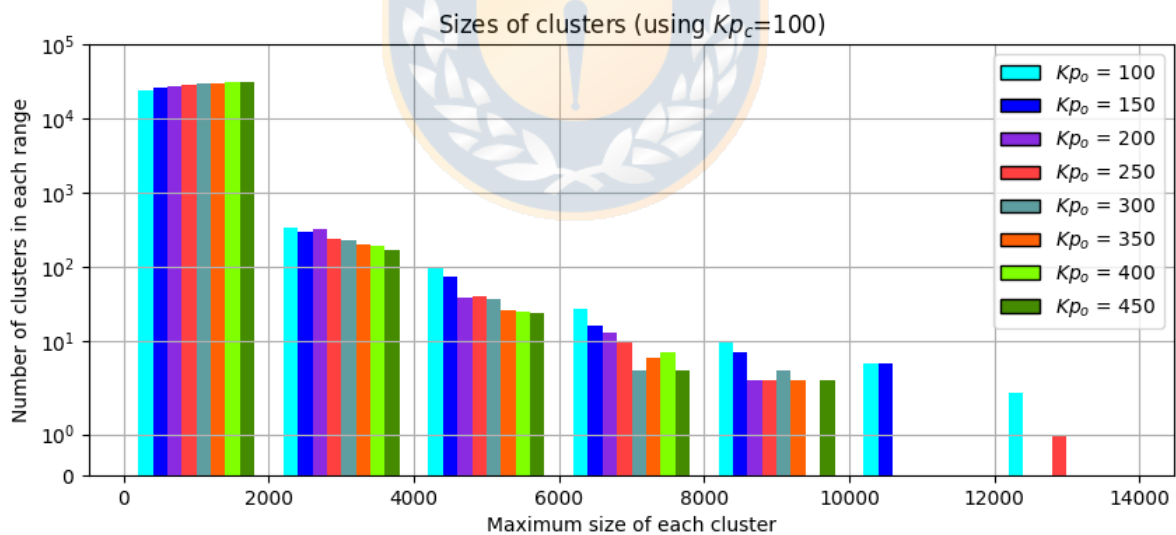


Figura 4.7: Histograma de tamaño de clústeres con 100 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función del tamaño de los clústeres utilizando $Kp_c = 100$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 100, 150, 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.

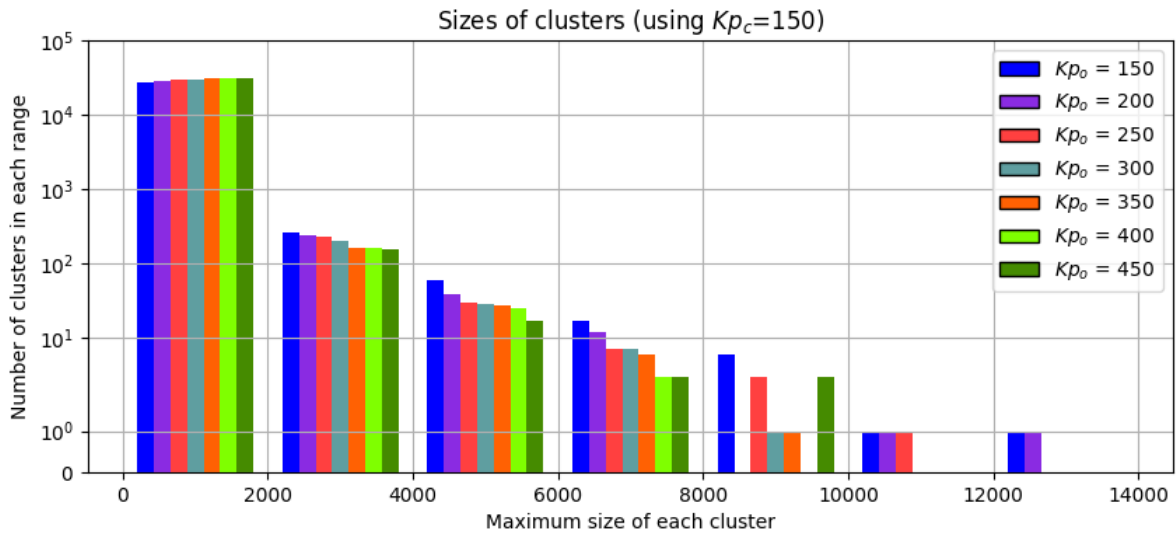


Figura 4.8: Histograma de tamaño de clústeres con 150 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función del tamaño de los clústeres utilizando $Kp_c = 150$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 150, 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.

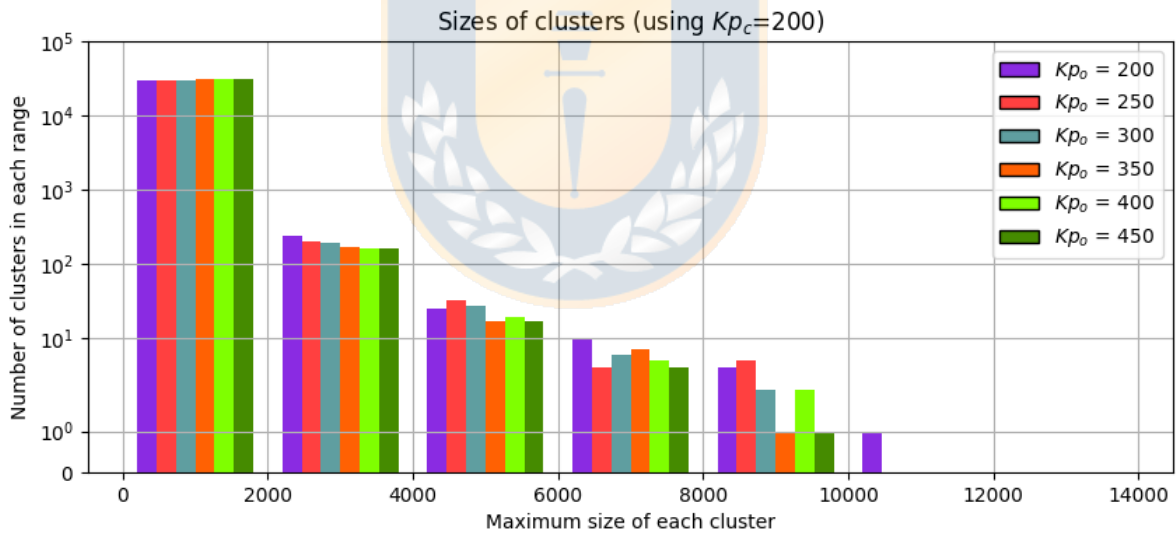


Figura 4.9: Histograma de tamaño de clústeres con 200 clústeres en los puntos del medio (Kp_c). Se muestra el número de clústeres en función del tamaño de los clústeres utilizando $Kp_c = 200$. El número de clústeres en los puntos extremos (Kp_o) toma valores de 200, 250, 300, 350, 400 y 450. Fuente: elaboración propia.

Como se muestra en las Figuras 4.7, 4.8 y 4.9, al igual que con las pruebas de

distancia máxima intraclúster, los resultados en función del tamaño de los clústeres son muy similares independientemente del Kp_c y Kp_o que se utilice. Sin embargo, analizando tanto los resultados en los histogramas de tamaño, junto con los histogramas de distancia intraclúster y el método Elbow, se han observado clústeres ligeramente mejores y más homogéneos utilizando los parámetros $Kp_c = 200$ y $Kp_o = 300$, así que, finalmente se ha optado por dicha configuración de valores de k.



- **Obtener el valor óptimo en los umbrales de reasignación y unión de clústeres.**

El objetivo de estas pruebas es encontrar el umbral de distancia máxima Euclidiana (d_{Rmax}) óptimo en la etapa de reasignación de clústeres (Figura 3.5-3 PASO 3), en la que se asignan los clústeres más pequeños a los clústeres más grandes para reducir el número de clústeres con cinco fibras o menos. También se evalúa el umbral de distancia máxima Euclidiana (d_{Mmax}) óptimo en la etapa de unión de clústeres (Figura 3.5-4 PASO 4), en la que se fusionan clústeres que están lo suficientemente cerca entre ellos.

Para realizar estas pruebas, se parte de la configuración del número de clústeres obtenido en las pruebas del apartado anterior, $K_{p_c} = 200$ y $K_{p_o} = 300$. Se generaron tres histogramas, a partir de varias ejecuciones con la siguiente configuración de parámetros: $thr_s = 4\text{mm}$, 6mm y 8mm , y, para cada uno de los valores de thr_s , se utilizaron los valores de $thr_j = 6\text{mm}$, 8mm , 10mm y 12mm . En los histogramas se muestra el número de clústeres en función de la medida de calidad más importante, la máxima distancia intraclúster (ver Figuras 4.10, 4.11 y 4.12).

Como se muestra en las Figuras 4.10, 4.11 y 4.12, los resultados de los histogramas son muy similares, es decir, hay muy poca diferencia en el número de clústeres según su distancia intraclúster utilizando diferentes configuraciones de thr_s y thr_j . Por ese motivo, se ha seleccionado la configuración de parámetros más conservadora, $thr_s = 6\text{mm}$ y $thr_j = 6\text{mm}$ para realizar los siguientes análisis, es decir, se eligen valores para los umbrales lo suficientemente pequeños como para que queden clústeres con poca distancia intraclúster.

En el siguiente apartado, se realiza la configuración de parámetros del método Quick-Bundles, para poder realizar una comparativa más justa en cuanto a la calidad de los clústeres y el tiempo de ejecución.

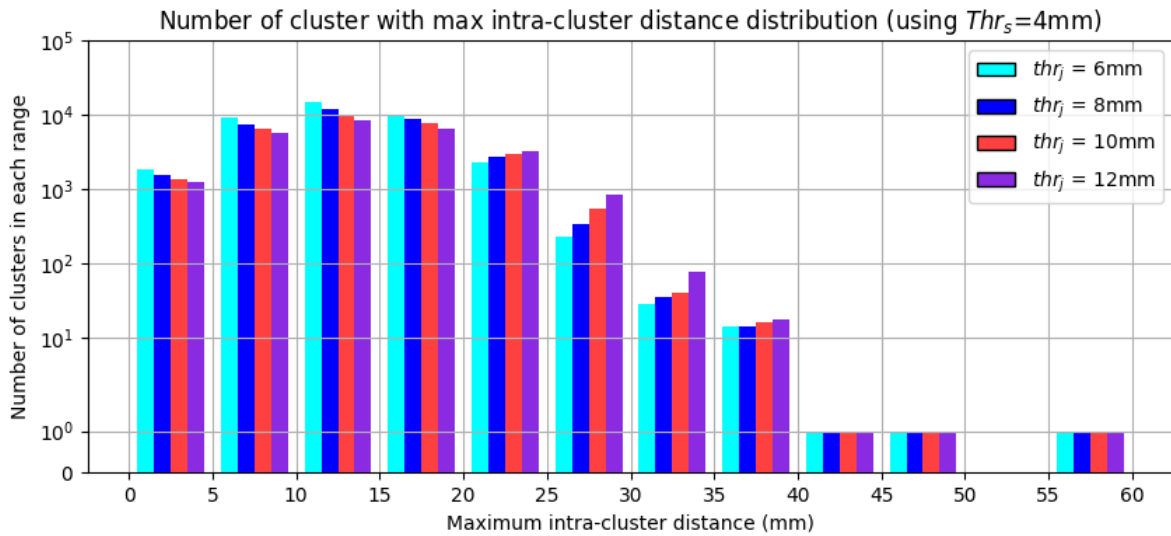


Figura 4.10: Histograma de distancia máxima intraclúster con $thr_s = 4mm$ y variando el valor de thr_j en 6mm, 8mm, 10 mm y 12mm. Fuente: elaboración propia.

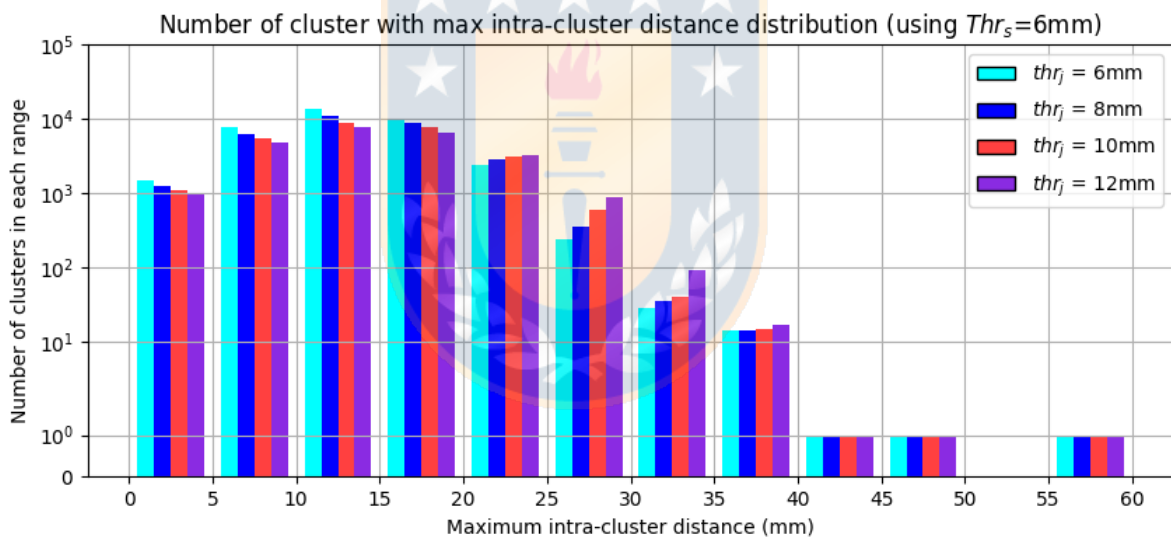


Figura 4.11: Histograma de distancia máxima intraclúster con $thr_s = 6mm$ y variando el valor de thr_j en 6mm, 8mm, 10 mm y 12mm. Fuente: elaboración propia.

4.3.2. Configuración de parámetros para el análisis cuantitativo de QuickBundles

El método QuickBundles (QB), presenta un parámetro configurable, el promedio mínimo de distancia directa e inversa entre cada par de fibras (minimum average

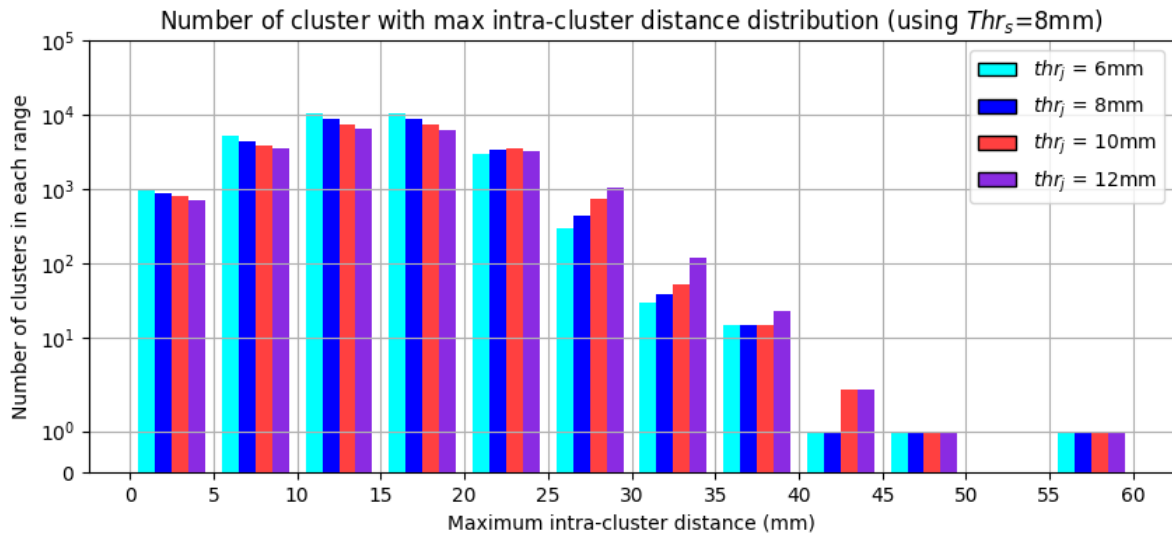


Figura 4.12: Histograma de distancia máxima intraclúster con $thr_s = 8\text{mm}$ y variando el valor de thr_j en 6mm, 8mm, 10 mm y 12mm. Fuente: elaboración propia.

direct-flip distance MDF), cuyo valor por defecto es de 10mm.

Debido a que el método QB acepta diferentes medidas de distancia, como primera prueba para este método, se ha implementado una nueva función de distancia, que en lugar de devolver el MDF , devuelve la distancia máxima entre cada par de fibras para que, de esta manera, QB funcione de la manera más similar posible a nuestro método. Sin embargo, la prueba ha sido fallida, debido a que la duración de la prueba se ha detenido a partir de las 73 horas de ejecución sin obtener ningún resultado para un conjunto de más de 3 millones de fibras.

Las siguientes pruebas del método QB, se han ejecutado con la medida de distancia por defecto presente en el método, la MDF , con valores 6, 8, 10 y 12mm respectivamente para 1 millón de fibras. Con dichos valores, se han generado dos histogramas, el primero muestra el número de clústeres en función de la distancia máxima intraclúster (ver Figura 4.13) y el segundo el número de clústeres en función del tamaño de los clústeres (ver Figura 4.14).

Como se muestra en la Figura 4.13, el número de clústeres y la distancia máxima

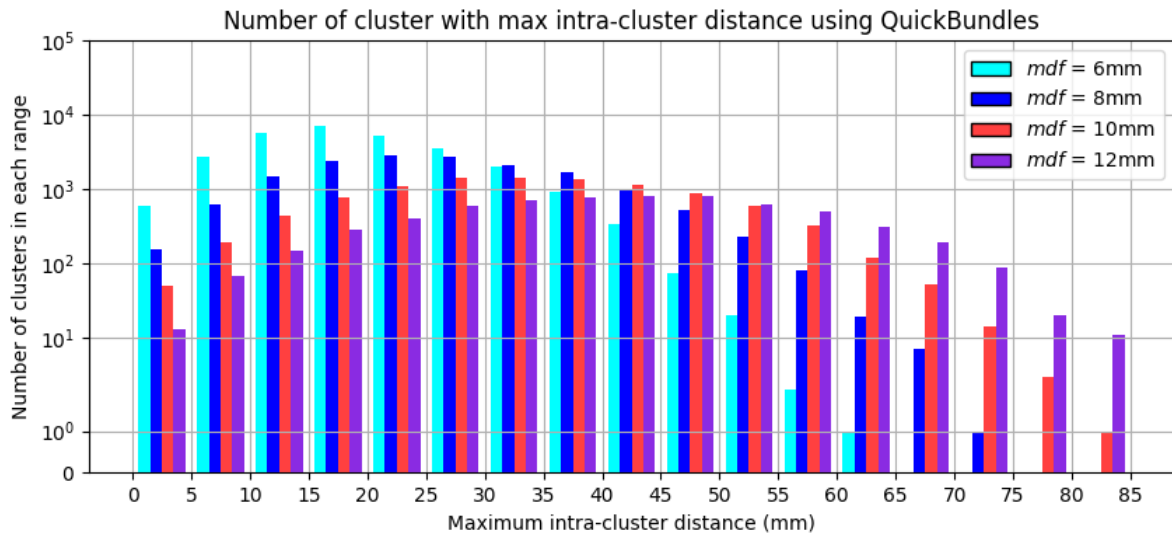


Figura 4.13: Histograma de distancia máxima intraclúster para el método QB. Se utiliza como valores de MDF 6, 8, 10 y 12mm respectivamente. Fuente: elaboración propia.

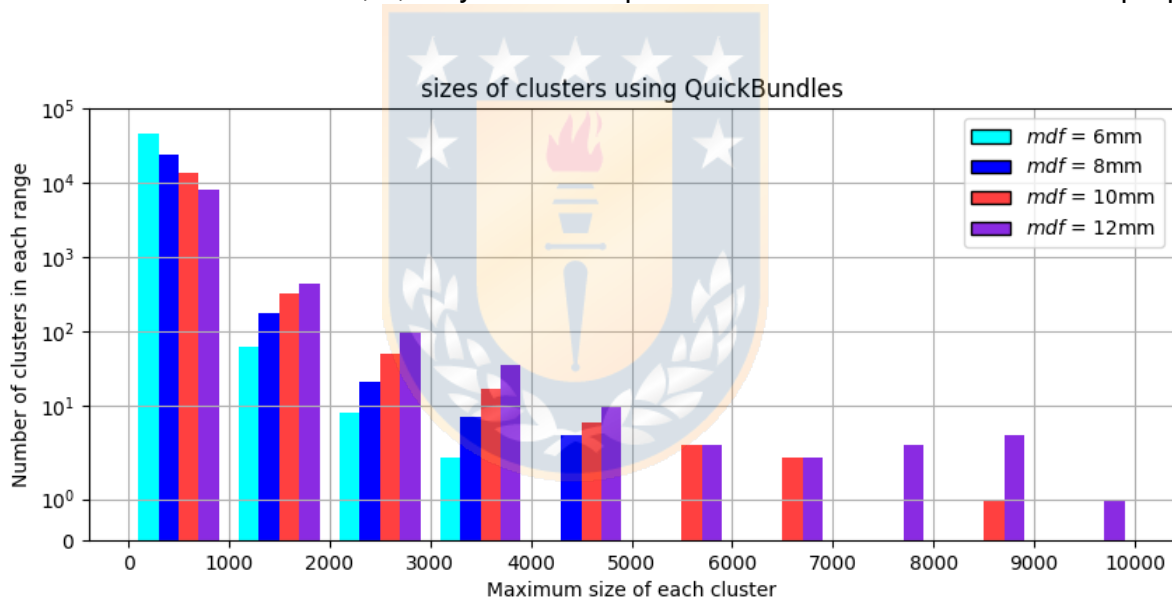


Figura 4.14: Histograma de tamaños de clúster para el método QB. Se utiliza como valores de MDF 6, 8, 10 y 12mm respectivamente. Fuente: elaboración propia.

intraclúster varía mucho en función de los valores de MDF , al ser una medida de calidad importante, se decide utilizar $MDF = 6mm$, que es la que conserva más clústeres con menor distancia intraclúster y que presenta los resultados más parecidos a nuestro método con los parámetros seleccionados. También se puede observar en la Figura 4.14, que hay muchos clústeres más pequeños con $MDF = 6mm$, esto es

debido a la restricción de distancia media entre cada par de fibras, ya que en QB, se van creando nuevos clústeres a medida que se realizan comparaciones.

Si se observa la Figura 4.13 con $MDF = 6\text{mm}$ y la Figura 4.11 de nuestro método con $thr_j = 6\text{mm}$, $Kp_c = 200$ y $Kp_o = 300$, se puede comprobar que los resultados en cuanto a distancia intraclúster son muy similares en ambos métodos. Sin embargo, hay que destacar un detalle muy importante en dichos resultados, en nuestro método, los clústeres con distancia máxima intraclúster a partir de 40mm, son unas pocas fibras ruidosas y en el método QB, esos mismos clústeres, son clústeres normales (esto se muestra detalladamente en el apartado de análisis cualitativo (4.3.3)).

Como conclusión, la calidad de los clústeres en nuestro método es superior que en el método de QB, debido a que los clústeres presentan menor distancia intraclúster. En cuanto al tamaño, nuestro método presenta clústeres más grandes que el método de QB y con una mayor variabilidad. Esto es normal debido a que en el cerebro hay clústeres de tamaños muy variables. En el apartado de análisis cualitativo (apartado 4.3.3) se puede observar que, visualmente, ambos métodos ofrecen muy buenos resultados.

4.3.3. Comparativa con QuickBundles mediante análisis cualitativo

En esta sección, se realiza la comparativa entre nuestro algoritmo y QB de forma cualitativa y con los parámetros previamente seleccionados. Para nuestro algoritmo se utilizan como parámetros: $Kp_c = 200$, $Kp_o = 300$, $thr_s = 6\text{mm}$ y $thr_j = 6\text{mm}$. Para el algoritmo QB, se utiliza la distancia MDF con los valores 6mm y 10mm. Como se ha visto anteriormente, QB con $MDF = 6\text{mm}$, es la configuración con más calidad de QB y resultados más similares a nuestro algoritmo, pero también es necesario realizar las pruebas con $MDF = 10$, debido a que es la medida por defecto y además le otorga una gran velocidad de ejecución al método QB.

Las cinco comparaciones más importantes son las siguientes: clusters más delgados (los 50 clústeres más delgados que presentan entre 2 y 5 fibras), clústeres más

gruesos (los 50 clústeres con más cantidad de fibras), clústeres de fibras cortas (200 clústeres de fibras cortas entre 30 y 60mm), clústeres de fibras largas (los 50 clústeres con las fibras más largas, a partir de 80mm) y los clústeres más separados (los clústeres con mayor distancia intraclúster). Las pruebas se ejecutaron sobre un sujeto con un millón de fibras de la base de datos ARCHI. A continuación, se describe detalladamente cada una de las comparaciones realizadas:

1. **Clústeres más delgados.** Se considera que los clústeres más delgados del resultado del clustering, son los que menos cantidad de fibras tienen, con la restricción de entre 2 y 5 fibras. Se ignoran los clústeres con una sola fibra porque nuestro algoritmo los reasigna o los elimina durante la etapa de reasignación.

En la Figura 4.15, se muestra el resultado de nuestro algoritmo y de QB con $MDF = 6\text{mm}$ y $MDF = 10\text{mm}$. En ambos algoritmos se muestra únicamente los 50 clústeres más delgados y, como se puede observar, con estas imágenes no se puede determinar visualmente ninguna diferencia en la calidad de los algoritmos.


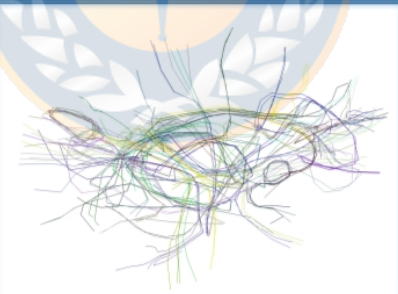

CLUSTERS MÁS DELGADOS		
NUESTRO ALGORITMO	QB. $MDF = 6\text{mm}$	QB. $MDF = 10\text{mm}$
		

Figura 4.15: Comparación de clústeres más delgados. En la comparación se muestra nuestro algoritmo con $Kp_c = 200$, $Kp_o = 300$, $thr_s = 6\text{mm}$, $thr_j = 6\text{mm}$, junto con QB con $MDF = 6\text{mm}$ y $MDF = 10\text{mm}$. Fuente: elaboración propia.

2. **Clústeres más gruesos.** Se considera que los clústeres más gruesos del resultado del clustering, son los que presentan mayor cantidad de fibras. De todos

los clústeres gruesos, se muestra únicamente los 50 clústeres más gruesos para que se pueda diferenciar bien la calidad.

En la Figura 4.16 se muestran las imágenes resultantes de ambos algoritmos, con las configuraciones previamente mencionadas. Además, se compara cada una de las tres vistas del cerebro (axial, coronal y sagital). Se puede observar que nuestro algoritmo y QB con $MDF = 6\text{mm}$ muestran clústeres de muy buena calidad, es decir, se ven uniformes y con los extremos “cerrados”, aunque los extremos de QB con $MDF = 6\text{mm}$ se ven un poco más “encrespados” o dispersos. En cuanto a QB con $MDF = 10\text{mm}$, se puede ver que los clústeres presentan una calidad más baja, demasiado grandes, con muchas fibras hacia todas las direcciones y con los extremos totalmente encrespados.

3. **Clústeres de fibras cortas.** Se considera que los clústeres de fibras cortas, son los que presentan fibras con una longitud de hasta 60mm. En la comparativa se muestran los 200 clústeres de fibras cortas más gruesos para comparar la calidad con mayor facilidad.

Esta comparativa es muy importante, debido a que, las fibras cortas del cerebro, son las más desconocidas, más variables y menos estudiadas. Además, las fibras cortas son mucho más numerosas que las conocidas fibras largas y, además, también conectan regiones del cerebro que controlan funcionalidades motoras, por lo que es muy importante su estudio.

En la Figura 4.17 se muestra la comparativa de fibras cortas para ambos algoritmos, el nuestro y QB con $MDF = 6\text{mm}$ y $MDF = 10\text{mm}$. Al igual que con los clústeres más gruesos, la calidad de QB con $MDF = 6\text{mm}$ y nuestro algoritmo son muy similares pero se observan algunos clústeres con extremos más dispersos. Pero QB con $MDF = 10\text{mm}$ presenta peor calidad, con clústeres demasiado anchos con los extremos “encrespados”.

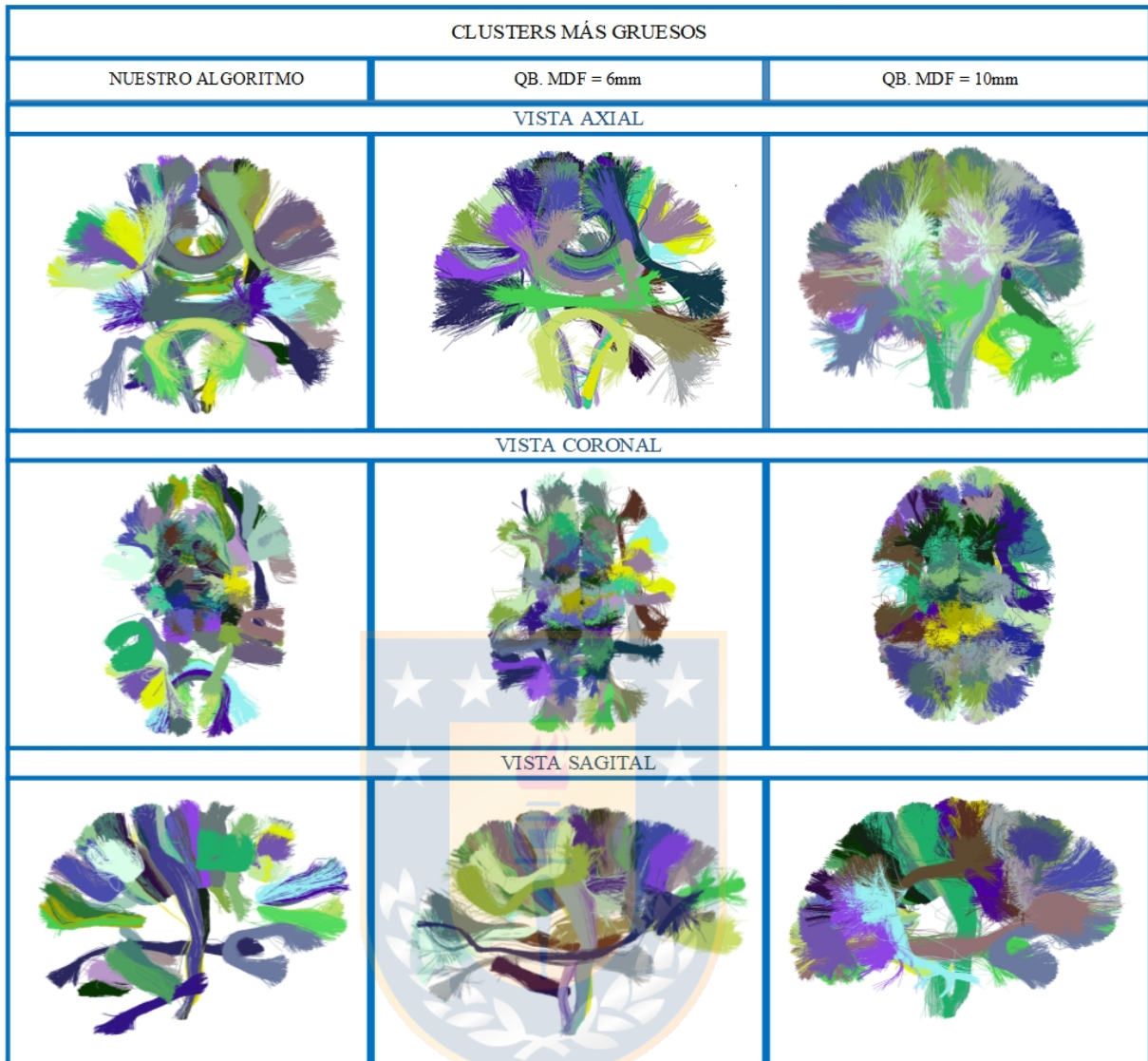


Figura 4.16: Comparación de clústeres más gruesos. En la comparación se muestra nuestro algoritmo con $Kp_c = 200$, $Kp_o = 300$, $thr_s = 6\text{mm}$, $thr_j = 6\text{mm}$, junto con QB con $MDF = 6\text{mm}$ y $MDF = 10\text{mm}$. Se comparan los tres tipos de vistas del cerebro, axial, coronal y sagital. Fuente: elaboración propia.

- Clústeres de fibras largas.** Se considera que los clústeres de fibras largas, son aquellos cuyas fibras presentan una longitud superior a 80mm. De todos los clústeres de fibras largas, se muestran únicamente los 50 clústeres de fibras más largas para permitir una mejor visualización.

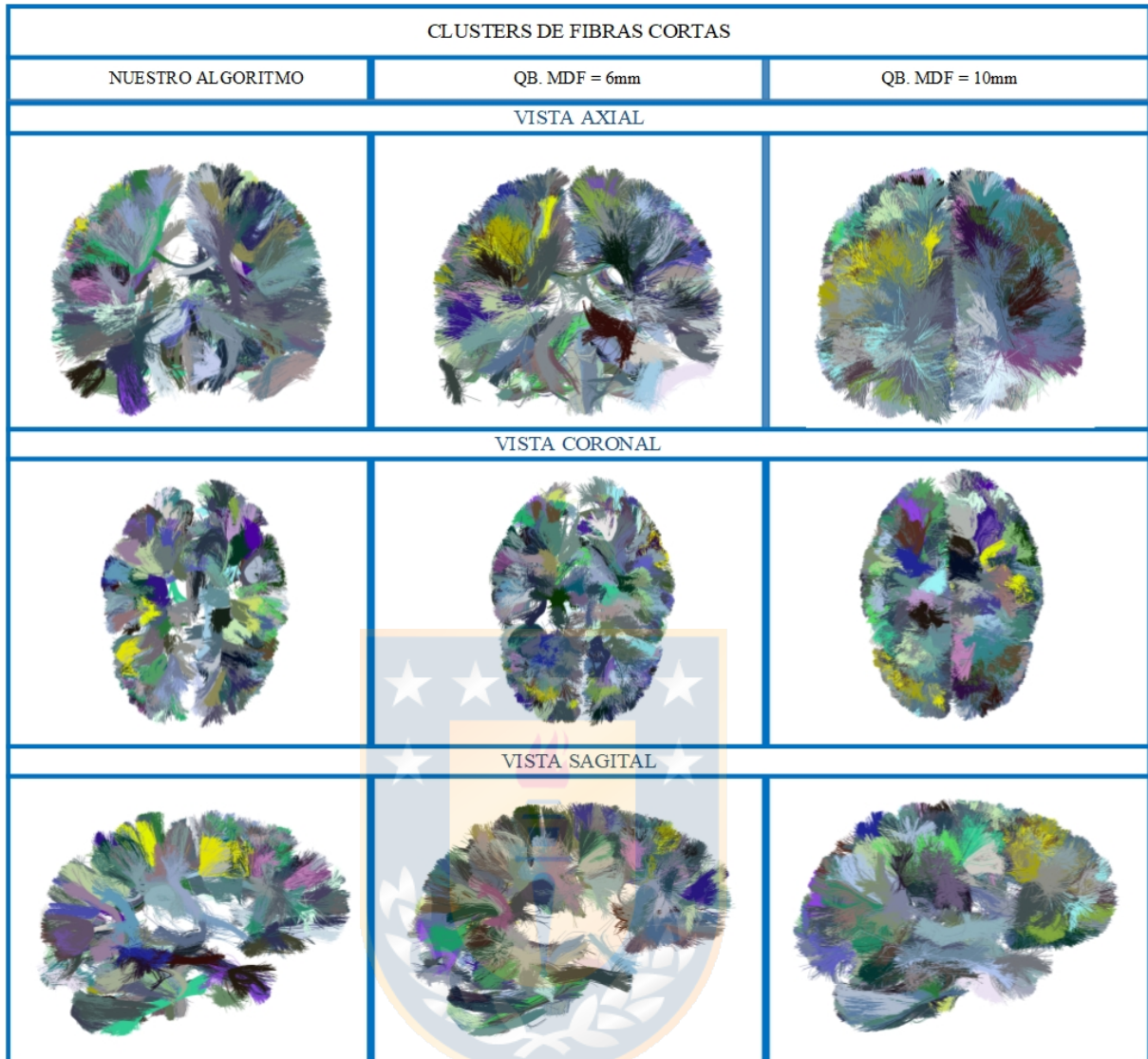


Figura 4.17: Comparación de clústeres de fibras cortas. Los clústeres contienen los 200 clústeres más gruesos con fibras de longitud entre 30mm y 60mm. En la comparación se muestra nuestro algoritmo con $Kp_c = 200$, $Kp_o = 300$, $thr_s = 6mm$, $thr_j = 6mm$, junto con QB con $MDF = 6mm$ y $MDF = 10mm$. Se comparan los tres tipos de vistas del cerebro, axial, coronal y sagital. Fuente: elaboración propia.

En la Figura 4.18 se puede observar la comparativa entre nuestro algoritmo y el algoritmo QB con $MDF = 6mm$ y $MDF = 10mm$, comparando las vistas axial, coronal y sagital. Al igual que en las pruebas anteriores, se aprecia que la calidad visual de los clústeres es buena con nuestro algoritmo y también con QB utilizando $MDF = 6mm$. Sin embargo, la calidad de los clústeres de QB con $MDF =$

10mm es mucho más baja, con clústeres más gruesos, mezclados entre ellos y los extremos de los clústeres están más encrespados.

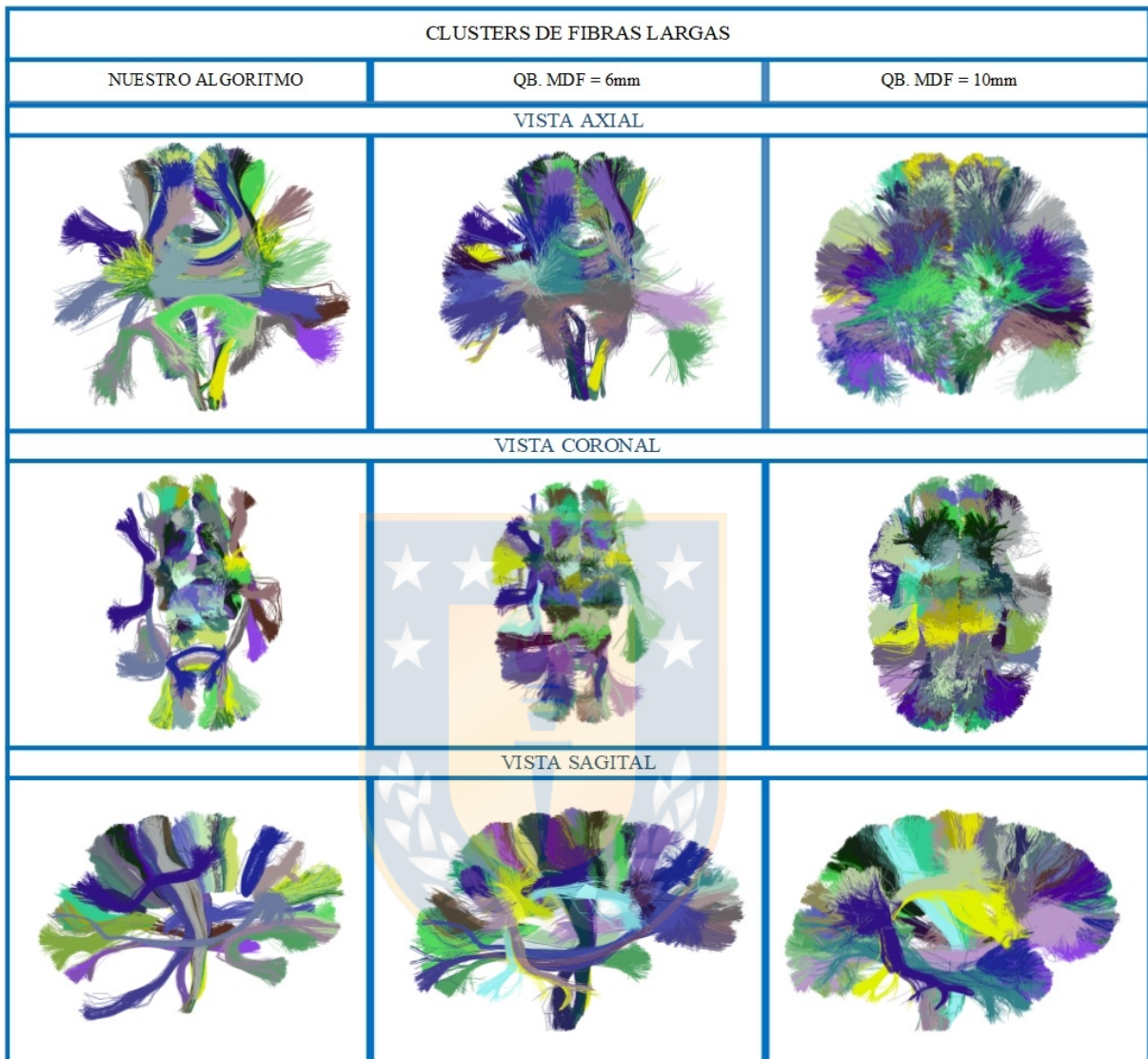


Figura 4.18: Comparación de clústeres de fibras largas. Los clústeres contienen los 50 clústeres más gruesos con fibras de longitud superior a 60mm. En la comparación se muestra nuestro algoritmo con $Kp_c = 200$, $Kp_o = 300$, $thr_s = 6mm$, $thr_j = 6mm$, junto con QB con $MDF = 6mm$ y $MDF = 10mm$. Se comparan los tres tipos de vistas del cerebro, axial, coronal y sagital. Fuente: elaboración propia.

5. Presencia de ruido en clústeres con gran distancia intraclúster.

Otra prueba importante que se ha realizado, es la visualización de los clústeres

con mayor distancia intraclúster. Como se ha podido ver en los histogramas del apartado 4.3.2 (ver Figuras 4.12, 4.13, 4.14), nuestro algoritmo presenta clústeres con una distancia hasta 60mm, el algoritmo QB con $MDF = 6\text{mm}$ también tiene clusters con hasta 60mm y QB con $MDF = 10\text{mm}$ hasta 80. Sin embargo, se ha verificado que en nuestro algoritmo los clústeres con una distancia intraclúster a partir de 40mm están formados por ruido de la tractografía. Las fibras ruidosas presentan problemas a la hora de unirse a los clústeres debido a que tienen formas atípicas y por eso se pueden visualizar.

En la Figura 4.19, se muestran los clústeres con mayor distancia intraclúster utilizando nuestro algoritmo, QB con $MDF = 6\text{mm}$ y QB con $MDF = 10\text{mm}$. Como se puede apreciar, los clústeres más separados que devuelve nuestro algoritmo, son únicamente dos clústeres pequeños con muy pocas fibras que presentan formas atípicas, es decir, es ruido en la tractografía. En cambio, en QB los clústeres más separados son clústeres normales, pero de mala calidad debido a que presentan una distancia intraclúster muy grande y sus fibras están más separadas y dispersas.




Nuestro algoritmo Distancia intra cluster > 40mm	QB con $MDF = 6\text{mm}$ Distancia intra cluster > 50mm	QB con $MDF = 10\text{mm}$ Distancia intra cluster > 70mm
		

Figura 4.19: Imágenes de los clústeres con mayor distancia intraclúster. En nuestro algoritmo se muestran con distancia intraclúster >40mm, en QB con $MDF = 6\text{mm}$ se muestran los clústeres con distancia >50mm y en QB con $MDF = 10\text{mm}$ los clústeres con distancia >70mm. Fuente: elaboración propia.

4.3.4. Comparativa en tiempo de ejecución con QuickBundles

En esta sección se realiza una comparativa en tiempo de ejecución de nuestro algoritmo con QuickBundles. Los parámetros de nuestro algoritmo son los mencionados en los apartados anteriores y se compara con QB con $MDF = 6\text{mm}$, que es la medida con la que presenta mayor calidad en los clústeres y también con $MDF = 10\text{mm}$, debido a que es la medida que utiliza QB por defecto y con la cual, su velocidad es mucho mayor.

Las pruebas de tiempo se han realizado utilizando sujetos de la base de datos ARCHI que tienen desde 330 mil hasta 2,7 millones de fibras. En la Tabla se muestra la comparativa en segundos de las ejecuciones con ambos algoritmos. Se puede observar que nuestro algoritmo es mucho más rápido que QB. Además, QB con $MDF = 6\text{mm}$ es extremadamente lento.

COMPARATIVA DE TIEMPOS EN SEGUNDOS			
Número de fibras	QB con $MDF = 10\text{mm}$	QB con $MDF = 6\text{mm}$	Nuestro algoritmo
330k	129,17s	393,24s	25,75s
659k	938,31s	3719,19s	55,83s
955k	1890,63s	7578,46s	87,3s
1296k	3619,18s	13274,39s	126,39s
1634k	6952,98s	21730,73s	138,11s
1945k	10294,75s	30816,33s	178,05s
2338k	14312,24s	50946,22s	366,47s
2729k	19861,15s	71243,69s	493,43s

Cuadro 4.1: Comparativa de tiempos con QB. Se muestran los resultados del tiempo de ejecución de nuestro algoritmo y QB con distancias 6 y 10mm en segundos. Se utilizaron sujetos de la base de datos ARCHI de hasta 2,7 millones de fibras. Fuente: elaboración propia.

En la Figura 4.20 se muestra la gran diferencia del tiempo de ejecución de ambos algoritmos y además se puede observar la tendencia del algoritmo QB, que es un

algoritmo lineal, sin embargo, es cuadrático en el peor de los casos.

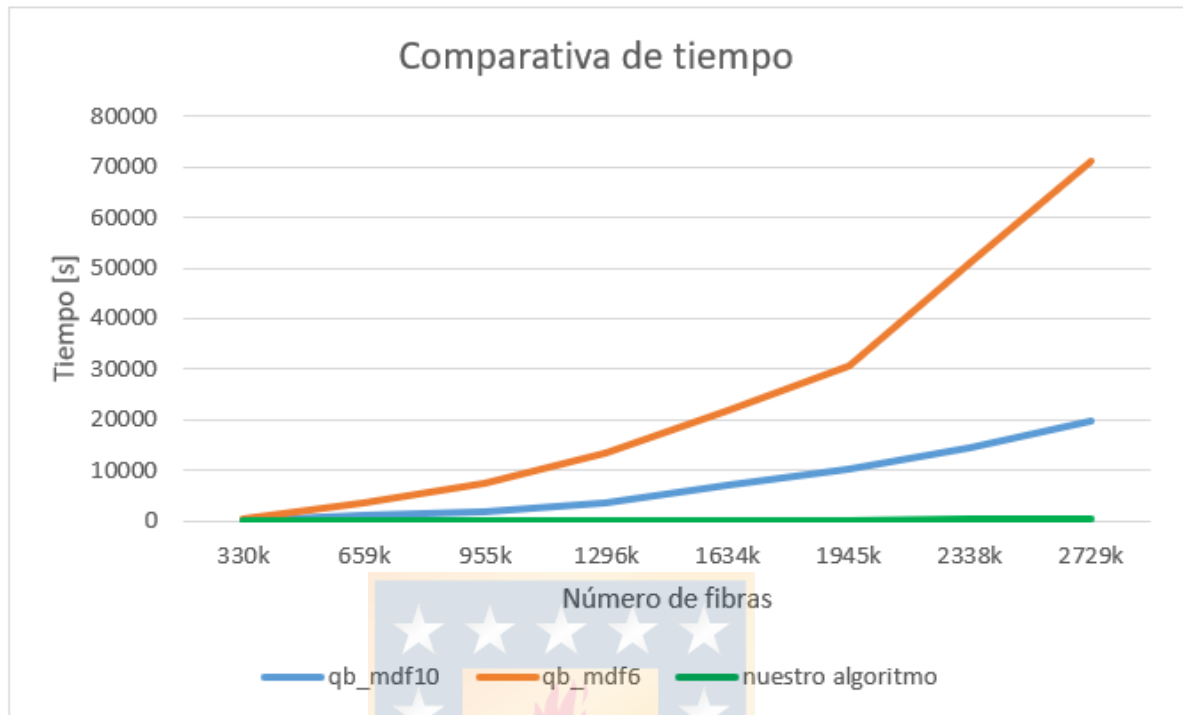


Figura 4.20: Gráfica comparativa de tiempos con QB. Se muestra la tendencia en tiempo de ejecución de los algoritmos en función del número de fibras de los sujetos, que tienen hasta 2.7 millones de fibras. Fuente: elaboración propia.

En la Figura 4.21 se muestra el desglose de tiempos para cada una de las etapas del algoritmo de clustering. A continuación, se explica en detalle cada uno de los resultados obtenidos en la gráfica en función de la etapa del algoritmo:

- Algoritmo K-means: A pesar de que el algoritmo utilizado para aplicar el clustering con K-means es lineal $\mathcal{O}(n)$, el tiempo devuelto para el sujeto de 1945 mil fibras fue muy bajo, ya que el tiempo de ejecución depende también del sujeto utilizado. Además, es importante destacar, que esta etapa es la que más tiempo tarda de todas.
- Etapa de mapeo: En esta etapa se mapean los clústeres de puntos a clústeres de fibras, es la etapa más rápida de todas y su complejidad es lineal $\mathcal{O}(n)$.

- Etapa de reasignación: La complejidad de esta etapa es cuadrática $\mathcal{O}(n^2)$, sin embargo, debido a que esta etapa está implementada en lenguaje C y paralelizada con OpenMP, es muy rápida.
- Unión de grupos: Para realizar la unión de grupos, se utiliza el algoritmo de clique, que es NP-hard, sin embargo, esta complejidad no se puede dar en nuestro conjunto de datos, debido a que se utilizan grafos muy pequeños y esparsos.

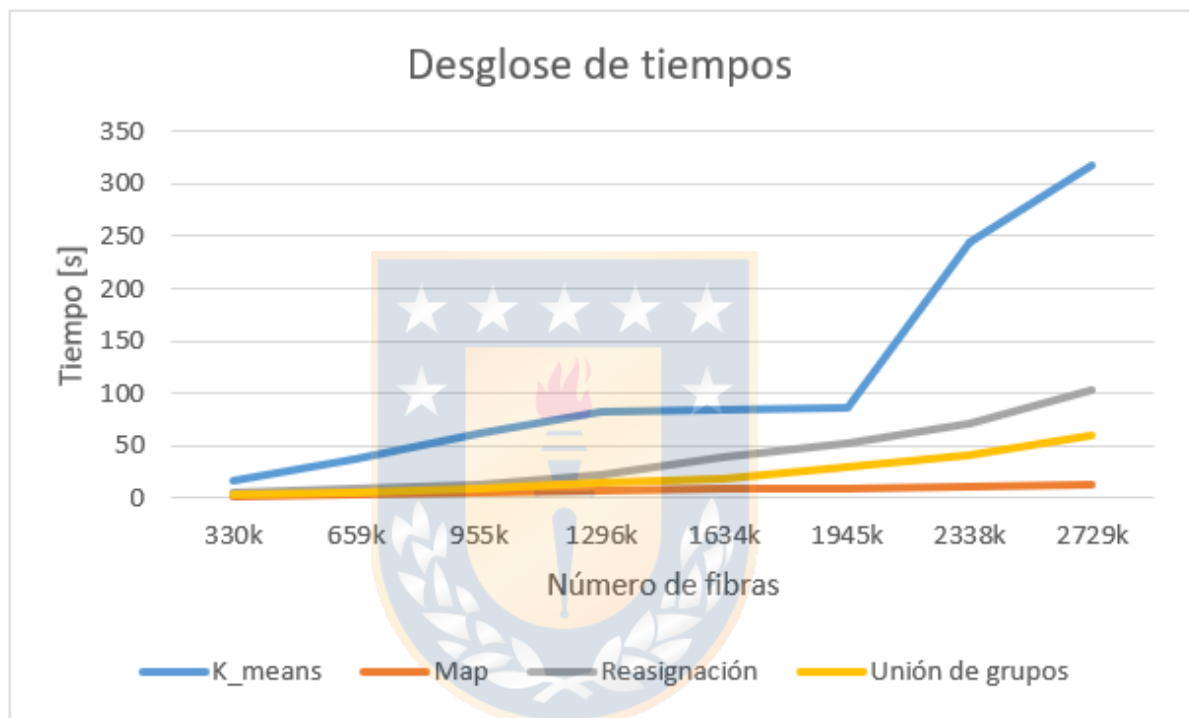


Figura 4.21: Desglose de tiempos de ejecución para cada una de las etapas del algoritmo de clustering: K-means, Map, Reasignación y unión de grupos. Las ejecuciones se realizan en sujetos de la base de datos ARCHI de hasta 2,7 millones de fibras. Fuente: elaboración propia.

Capítulo 5

CONCLUSIONES

5.1. CONCLUSIONES GENERALES

En la presente tesis se ha presentado un nuevo algoritmo paralelo para la segmentación de fibras de WM. En comparación con una versión anterior del algoritmo, la nueva versión reduce el tiempo de segmentación de un conjunto de datos de 4,145,000 fibras de 14 minutos a 6 minutos, equivalente a un aumento de velocidad de 2,34. Además, para el mismo número de fibras, la versión anterior consume 1.25 veces más memoria.

El nuevo algoritmo tiene una complejidad espacial lineal, lo que le permite cargar todo el conjunto de datos y explotar más paralelismo que la versión original, incluso en computadoras con memoria limitada y espacio de intercambio. Además, es más escalable, lo que abre la posibilidad de aplicarlo a conjuntos de datos de tractografía más grandes y atlas más grandes, en particular, los obtenidos de fibras de materia blanca superficial (SWM), basadas en la tractografía probabilística, que puede alcanzar varios millones de fibras. Para conjuntos de tamaño pequeño puede permitir interactividad.

Por otra parte, realiza una mejor clasificación, etiquetando las fibras solo con el fascículo más cercano. Adicionalmente, la nueva implementación en C++ permite extensiones y mantenimiento futuros más fáciles del código. Además, aplicaciones en planificación de neurocirugía y comparación de fascículos entre grupos de sujetos.

El otro método implementado es un algoritmo innovador para realizar clustering de fibras de la materia blanca del cerebro. Como se ha visto en las pruebas, la calidad de

los clústeres obtenidos es muy buena y es muy similar al resultado con el algoritmo QB con $MDF = 6$ mm, aunque ligeramente mejor, ya que los clústeres obtenidos tienen menor distancia intraclúster y, por lo tanto, sus fibras están más juntas y sus extremos menos “encrespados”.

Aparte de la calidad de los clústeres, otra mejora muy notable es el tiempo de ejecución, ya que, con un sujeto de 2,7 millones de fibras, nuestro algoritmo tarda 8,22 minutos. En cambio, el mejor algoritmo de clustering presente en el estado del arte, QuickBundles (QB), tarda con el mismo sujeto 19,7 horas utilizando su mejor configuración de calidad ($MDF = 6$ mm). Es cierto que QB con su distancia por defecto ($MDF = 10$ mm), es un poco más rápido que con la otra, ya que tarda 5,5 horas, pero la pérdida de calidad también es muy significativa.

Por último, es importante destacar que nuestro algoritmo de clustering, al estar dividido en varias etapas, es muy factible como trabajo futuro optimizar las etapas más lentas, para que la velocidad de clustering sea mucho menor. Esto podría ser útil a la hora de integrarlo con aplicaciones de visualización, para tratar de realizar clustering en un tiempo cercano al tiempo real, y poder visualizar la estructura de la materia blanca de forma rápida en datos masivos de tractografía.

5.2. TRABAJO FUTURO

Como trabajo futuro se propone realizar el método de clustering en la GPU. Para realizar la primera parte del proceso de clustering por puntos, se utilizará un algoritmo paralelo de K-means implementado en el lenguaje de programación C sobre CUDA obtenido del repositorio de Giuroiu (2013). En concreto, se utilizará para realizar K-means sobre los puntos de las fibras de forma paralela.

5.2.1. Tareas pendientes para completar el método de clustering

El algoritmo implementado en CUDA, únicamente realiza clustering sobre los puntos de las fibras, en concreto, está optimizado para trabajar sobre un único punto.

Fases	Tiempo en segundos					
	Punto 1		Punto 4			Punto 11
	CUDA	MBKM	CUDA	MBKM	CUDA	MBKM
1. clústeres de puntos	0,06	2,43	0,07	4,13	0,05	3,72
2. clústeres preliminares	-	0,28	-	0,28	-	0,27
3. Reasignación de clústeres	-	6,36	-	6,42	-	6,37
4. Fusión de clústeres	-	8,21	-	8,27	-	8,21

Cuadro 5.1: Tiempo de ejecución en la primera fase de los algoritmos MiniBatch K-means y CUDA K-means. Fuente: elaboración propia.

Se podría ejecutar el algoritmo secuencialmente, sin embargo no se estarían aprovechando al máximo los recursos de la GPU. A continuación se enumeran algunas de las tareas que faltan por realizar para finalizar el método completo de clustering:

- Optimizar el algoritmo de CUDA K-means para trabajar con varios puntos de fibras al mismo tiempo (clústeres de puntos).
- Implementar los clústeres preliminares en CUDA.
- Implementar la reasignación de clústeres en CUDA.
- Implementar la fusión de clústeres en CUDA.
- Mejorar la calidad de los clústeres: Como se puede ver en la Tabla 5.1, los tiempos de ejecución en CUDA son muy bajos. El algoritmo de Sanchez et al. (2018) realiza clustering sobre los puntos 1, 4, 11, 17 y 21 sobre streamlines divididos en 21 puntos equidistantes. Si se aumenta el número de puntos, mejora la calidad de los clústeres finales. En el caso del algoritmo de CUDA, se podría estudiar la relación entre el tiempo de ejecución y el número de puntos utilizados para mejorar la calidad de los clústeres finales.

5.2.2. Inclusión del algoritmo de Clustering en un visualizador

También como trabajo futuro se propone incluir el algoritmo realizado en la presente tesis, en un visualizador para poder ver el resultado del clustering. Además, debido al poco tiempo de ejecución que se demora, es posible que este método sea de utilidad

para neurólogos y neuroanatomistas para realizar el estudio de las fibras de la materia blanca.

5.3. PUBLICACIONES

Conferencias internacionales

El método de segmentación de fibras se ha enviado a dos conferencias, una internacional, International Symposium on Biomedical Imaging (ISBI) a realizarse en Venecia en el año 2019 entre los días 8 y 11 de abril. Los datos del paper son:

- Título: PARALLEL OPTIMIZATION OF FIBER BUNDLE SEGMENTATION FOR MASSIVE TRACTOGRAPHY DATASETS.
- Autores: Andrea Vázquez, Narciso López-López, Nicole Labra, Miguel Figueroa, Cyril Poupon, Jean-François Mangin, Cecilia Hernández and Pamela Guevara.
- Instituciones: Universidad de Concepción, Concepción, Chile
Universidade de A Coruña, Coruña, España.
Neurospin, CEA, Gif-Sur-Yvette, France.

Este trabajo fue aceptado y se encontrará en los proceedings de ISBI 2019.

Conferencias nacionales

La otra conferencia a la que se envió el método de segmentación, fue la realizada en Valdivia en el año 2018, en concreto en INGELECTRA 2018 entre los días 6 y 7 de diciembre. Los datos del paper son:

- Título: OPTIMIZACIÓN PARALELA PARA LA SEGMENTACIÓN DE FASCÍCULOS DE FIBRAS EN CONJUNTOS MASIVOS DE DATOS DE TRACTOGRAFÍA.
- Autores: Andrea Vázquez, Narciso López-López, Miguel Figueroa, Cecilia Hernández y Pamela Guevara.

- Instituciones: Universidad de Concepción, Concepción, Chile
Universidade de A Coruña, Coruña, España.

Este paper fue aceptado y se presentó en varias sesiones de pósters durante IN-GELECTRA 2018, llevándose el primer premio en la conferencia al mejor trabajo de Postgrado.

Paper ISI

El método principal de este trabajo se enviará a la revista Nature Communications (como alternativa se enviará a NeuroImage). El contenido es el mismo al apartado del método de Clustering de fibras y sus resultados correspondientes.

- Título: CENTELLA: An efficient method for Clustering White Matter Fibers.
- Autores: Andrea Vázquez, Alexis Sánchez, Narciso López-López, Cyril Poupon, Jean-François Mangin, Cecilia Hernández and Pamela Guevara.
- Instituciones: Universidad de Concepción, Concepción, Chile
Universidade de A Coruña, Coruña, España.
Neurospin, CEA, Gif-Sur-Yvette, France.

Este paper se encuentra en su fase final de redacción y revisión.

Colaboraciones

Se envió un abstract a Organization for Human Brain Mapping (OHBM). Conferencia a realizarse en Roma (Italia) en el 2019 entre los días 9 y 13 de junio.

En este abstract se ha presentado una herramientas para la manipulación y visualización de las fibras de la materia blanca cerebrales, que utiliza el algoritmo de segmentación desarrollado en este trabajo.

- Título: FiberVis: a tool for a fast fiber tractography visualization and segmentation.

- Autores: Ignacio Osorio, Danilo Bonometti, Diego Carrasco, Andrea Vázquez, Narciso López-López, Cyril Poupon, Jean-François Mangin, Pamela Guevara.
- Instituciones: Universidad de Concepción, Concepción, Chile Neurospin, CEA, Gif-Sur-Yvette, France.

En la actualidad (26/03/2019) el trabajo se encuentra enviado a la espera de contestación por la organización.



Bibliografía

- Ascher, U. M. and Petzold, L. R. (1998). *Computer methods for ordinary differential equations and differential-algebraic equations*, volume 61. Siam.
- Basser, P. J., Pajevic, S., Pierpaoli, C., Duda, J., and Aldroubi, A. (2000). In vivo fiber tractography using dt-mri data. *Magnetic resonance in medicine*, 44(4):625–632.
- Bonometti, D., Barraza, Wallace, I. O., Duclap, D., Lebois, A., Poupon, C., Mangin, J.-F., and Guevara, P. (2015). A fast tractography visualization tool using modern opengl. In *Conference of the IEEE Engineering in Medicine and Biology Society*.
- Catani, M., Dell’Acqua, F., Vergani, F., and et al. (2012). Short frontal lobe connections of the human brain. *cortex*, 48(2):273–291.
- Catani, M., Howard, R. J., Pajevic, S., and Jones, D. K. (2002). Virtual in vivo interactive dissection of white matter fasciculi in the human brain. *Neuroimage*, 17(1):77–94.
- Eppstein, D. and Strash, D. (2011). Listing all maximal cliques in large sparse real-world graphs. In *International Symposium on Experimental Algorithms*, pages 364–375. Springer.
- Filippi, M., Rocca, M. A., Ciccarelli, O., De Stefano, N., Evangelou, N., Kappos, L., Rovira, A., Sastre-Garriga, J., Tintorè, M., Frederiksen, J. L., et al. (2016). Mri criteria for the diagnosis of multiple sclerosis: Magnims consensus guidelines. *The Lancet Neurology*, 15(3):292–303.
- Frisoni, G. B., Fox, N. C., Jack Jr, C. R., Scheltens, P., and Thompson, P. M. (2010). The clinical use of structural mri in alzheimer disease. *Nature Reviews Neurology*, 6(2):67.
- Garyfallidis, E., Brett, M., Amirbekian, B., Rokem, A., Van Der Walt, S., Descoteaux, M., and Nimmo-Smith, I. (2014). Dipy, a library for the analysis of diffusion mri data. *Frontiers in neuroinformatics*, 8:8.
- Garyfallidis, E., Brett, M., Correia, M. M., Williams, G. B., and Nimmo-Smith, I. (2012). Quickbundles, a method for tractography simplification. *Frontiers in neuroscience*, 6:175.
- Giuroiu, S. (2013). A cuda implementation of the k-means clustering algorithm. <https://github.com/serban/kmeans>.

- Guevara, M., Osorio, I., Bonometti, D., Duclap, D., Poupon, C., Mangin, J., and Guevara, P. (2015). ifiber: A brain tract visualizer for android devices. In *Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON), 2015 CHILEAN Conference on*, pages 245–250. IEEE.
- Guevara, M., Román, C., Houenou, J., Duclap, D., Poupon, C., Mangin, J. F., and Guevara, P. (2017). Reproducibility of superficial white matter tracts using diffusion-weighted imaging tractography. *NeuroImage*, 147:703–725.
- Guevara, P., Duclap, D., Poupon, C., et al. (2012a). Automatic fiber bundle segmentation in massive tractography datasets using a multi-subject bundle atlas. *NeuroImage*, 61(4):1083–1099.
- Guevara, P., Duclap, D., Poupon, C., Marrakchi-Kacem, L., Fillard, P., Le Bihan, D., Leboyer, M., Houenou, J., and Mangin, J.-F. (2012b). Automatic fiber bundle segmentation in massive tractography datasets using a multi-subject bundle atlas. *Neuroimage*, 61(4):1083–1099.
- Guevara, P., Poupon, C., Rivière, D., Cointepas, Y., Descoteaux, M., Thirion, B., and Mangin, J.-F. (2011a). Robust clustering of massive tractography datasets. *NeuroImage*, 54(3):1975–1993.
- Guevara, P., Poupon, C., Rivière, D., and et al. (2011b). Robust clustering of massive tractography datasets. *NeuroImage*, 54(3):1975–1993.
- Haacke, E. M., Brown, R. W., Thompson, M. R., Venkatesan, R., et al. (1999). *Magnetic resonance imaging: physical principles and sequence design*, volume 82. Wiley-liss New York:.
- Hardvard (2018). The brain is a series of tubes. <http://sitn.hms.harvard.edu/art/2016/brain-series-tubes/>. Último acceso: 01-05-2018.
- Jin, Y., Shi, Y., Zhan, L., Gutman, B. A., de Zubicaray, G. I., McMahon, K. L., Wright, M. J., Toga, A. W., and Thompson, P. M. (2014). Automatic clustering of white matter fibers in brain diffusion mri with an application to genetics. *NeuroImage*, 100:75–90.
- Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer.
- Kodinariya, T. M. and Makwana, P. R. (2013). Review on determining number of cluster in k-means clustering. *International Journal*, 1(6):90–95.
- Labra, N., Guevara, P., Duclap, D., and et al. (2017). Fast automatic segmentation of white matter streamlines based on a multi-subject bundle atlas. *Neuroinformatics*, 15(1):71–86.

- Le Bihan, D. and Lima, M. (2015). Diffusion magnetic resonance imaging: what water tells us about biological tissues. *PLoS Biology*, 13(7):e1002203.
- Le Bihan, D., Mangin, J. F., Poupon, C., and et al. (2001). Diffusion tensor imaging: concepts and applications. *J Magn Reson Imaging*, 13(4):534–546.
- Mangin, J.-F., Régis, J., and Frouin, V. (1996). Shape bottlenecks and conservative flow systems [medical image analysis]. In *Mathematical Methods in Biomedical Image Analysis, 1996., Proceedings of the Workshop on*, pages 319–328. IEEE.
- Merboldt, K.-D., Hanicke, W., and Frahm, J. (1985). Self-diffusion nmr imaging using stimulated echoes. *Journal of Magnetic Resonance (1969)*, 64(3):479–486.
- O'Donnell, L. and Westin, C.-F. (2007). Automatic tractography segmentation using a high-dimensional white matter atlas. *IEEE Transactions on Medical Imaging*, 26(11):1562–1575.
- O'Donnell, L., Kubicki, M., Shenton, M. E., Dreusicke, M. H., Grimson, W. E. L., and Westin, C.-F. (2006). A method for clustering white matter fiber tracts. *American Journal of Neuroradiology*, 27(5):1032–1036.
- Presseau, C., Jodoin, P.-M., Houde, J.-C., and Descoteaux, M. (2015). A new compression format for fiber tracking datasets. *NeuroImage*, 109:73–83.
- Rheault, F., Houde, J.-C., and Descoteaux, M. (2017). Visualization, interaction and tractometry: Dealing with millions of streamlines from diffusion mri tractography. *Frontiers in neuroinformatics*, 11:42.
- Román, C., Guevara, M., Valenzuela, R., and et al. (2017). Clustering of whole-brain white matter short association bundles using hardi data. *Frontiers in neuroinformatics*, 11:73.
- Ros, C., Güllmar, D., Stenzel, M., and et al. (2013). Atlas-guided cluster analysis of large tractography datasets. *PloS one*, 8(12):e83847.
- Sanchez, A., Hernández, C., Poupon, C., Mangin, J.-F., and Guevara, P. (2018). Clustering of tractography datasets based on streamline point distribution. In *International Society of Magnetic Resonance in Medicine conference*. ISMRM 2018.
- Schmitt, B., Lebois, A., Duclap, D., and et al. (2012a). Connect/archi: an open database to infer atlases of the human brain connectivity. In *ESMRMB conference*.
- Schmitt, B., Lebois, A., Duclap, D., Guevara, P., Poupon, F., Rivière, D., Cointepas, Y., LeBihan, D., Mangin, J., and Poupon, C. (2012b). Connect/archi: an open database to infer atlases of the human brain connectivity. *ESMRMB*, 272:2012.
- Sculley, D. (2010). Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178. ACM.

- Sipser, M. (2006). *Introduction to the Theory of Computation*, volume 2. Thomson Course Technology Boston.
- Steinbach, M., Karypis, G., Kumar, V., et al. (2000). A comparison of document clustering techniques. In *KDD workshop on text mining*, volume 400, pages 525–526. Boston.
- Visser, E., Nijhuis, E. H., Buitelaar, J. K., and Zwiers, M. P. (2011). Partition-based mass clustering of tractography streamlines. *Neuroimage*, 54(1):303–312.
- Wassermann, D., Bloy, L., Kanterakis, E., and et al. (2010). Unsupervised white matter fiber clustering and tract probability map generation: Applications of a gaussian process framework for white matter fibers. *Neuroimage*, 51:228–241.
- Wassermann, D., Makris, N., Rathi, Y., and et al. (2016). The White Matter Query Language: A novel approach for describing human white matter anatomy. *Brain Struct Funct*, 221(9):4705–4721.
- Woeginger, G. J. (2003). Exact algorithms for np-hard problems: A survey. In *Combinatorial Optimization—Eureka, You Shrink!*, pages 185–207. Springer.
- Yao, N., Winkler, A. M., Barrett, J., Book, G. A., Beetham, T., Horseman, R., Leach, O., Hodgson, K., Knowles, E. E., Mathias, S., et al. (2017). Inferring pathobiology from structural mri in schizophrenia and bipolar disorder: Modeling head motion and neuroanatomical specificity. *Human brain mapping*, 38(8):3757–3770.
- Yeatman, J. D., Richie-Halford, A., Smith, J. K., Keshavan, A., and Rokem, A. (2018). A browser-based tool for visualization and analysis of diffusion mri data. *Nature communications*, 9(1):940.
- Zhang, Y., Zhang, J., Oishi, K., and et al. (2010). Atlas-guided tract reconstruction for automated and comprehensive examination of the white matter anatomy. *Neuroimage*, 52(4):1289 – 1301.