Universitat Autònoma de Barcelona

escola d'enginyeria

3899 BIOINFORMÁTICA: LAYOUT DE GRAFOS INTERACTIVOS PARA MATRICES DE EXPRESIÓN GÉNICA DE GRAN VOLUMEN

Memoria del Proyecto Final de Carrera de Ingeniería en Informática y Matemáticas realizado por Raquel Guardia Villalba y dirigido por Mario Huerta y Jordi Gonzàlez i Sabaté

Bellaterra, 05 de Setiembre de 2011

El sotasignat, Jordi Gonzàlez i Sabaté

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat

sota la seva direcció per Raquel Guardia Villalba.

I per tal que consti firma la present.

Signat:

Bellaterra, 22 de Junio de 2011

Universitat Autònoma de Barcelona

Escola Tècnica Superior d'Enginyeria

El sotasignat, Mario Huerta

de l'empresa, Institut de Biotecnologia i de Biomedicina de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat a l'empresa sota la seva supervisió mitjançant conveni amb la Universitat Autònoma de Barcelona.

Així mateix, l'empresa en té coneixement i dóna el vist-i-plau al contingut que es detalla en aquesta memòria.

Signat:

Bellaterra, 22 de Junio de 2011

Tabla de contenido

1. Introducción

1.1 Motivación personal

La siguiente reflexión refleja lo que a muchos nos sucede en un momento determinado de nuestra vida. Ese momento en que te das cuenta de que no sabes nada de un tema determinado. En ese momento hay dos opciones: indiferencia o curiosidad. Pues bien, a mí me picó el gusanillo de la curiosidad poco antes de acabar mi formación universitaria.

En los últimos años de carrera, una vez conocidos diversos lenguajes de programación, cómo tratar diversos aspectos de la informática e incluso ciertos entresijos que esconden las matemáticas, sentí que era necesario aplicar toda esa base de conocimiento a algún campo práctico. Necesitaba saber el sentido de haber procesado tanta información y, sobretodo, darle utilidad en un nuevo campo y, ¿qué mejor campo que el de nuestra propia salud y la de los que nos rodean?

Por este motivo me interesé por la bioinformática y decidí realizar este proyecto en el Institut de Biotecnologia i BioMedicina (IBB) de la Universidad Autónoma de Barcelona. Esta entidad, debido a su naturaleza multidisciplinar, plantea soluciones que combinan los avances y técnicas de Bioinformática, Genómica, Biología celular y otras ciencias frente a los importantes desafíos y problemas biológicos que se plantean en sus investigaciones.

Por otra parte, cierto es que dicha entidad ya contaba con aplicaciones que permitían trabajar con datos de un cierto orden. No obstante, estamos en unos tiempos en los cuales la rapidez premia y los datos se generan a un ritmo mayor que como son procesados, por lo que es posible que si trabajas con más datos a la vez, contando para ello con una aplicación de fácil manejo y que pone a tu alcance todas las soluciones que puedas necesitar, encuentres hechos y resultados que no podrías haber hallado trabajando con menos información.

Finalmente, comentar que me enorgullece ver como he conseguido satisfacer las necesidades que se me expusieron, un caso real con una aplicación real y con gran utilidad médica. Un problema no solo mío, sino de toda la comunidad científica en el campo de la bioinformática. Y esta “necesidad científica” la he satisfecho aplicando los conocimientos que he adquirido en el transcurso de la carrera (más algunos nuevos adquiridos expresamente para el proyecto). De esta manera, puedo apreciar como lo aprendido en la universidad es útil una vez finalizada esta etapa de mi formación.

1.2 Estado del arte

La bioinformática es la ciencia dedicada al estudio de los fenómenos biológicos desde un punto de vista computacional con el objetivo de ofrecer métodos robustos para la comprensión, simulación y predicción de comportamientos biológicos observados en los seres vivos. De esta manera los principales esfuerzos de investigación en este campo incluyen el alineamiento de secuencias, la predicción de genes, el alineamiento estructural de proteínas, la predicción de estructura de proteínas, el estudio de la expresión génica o las interacciones proteína-proteína entre otros.

Una constante en proyectos de este tipo es el uso de herramientas matemáticas para extraer información útil de datos producidos por técnicas experimentales de alta productividad, como la tecnología de microarrays.

Los genes al expresarse, sintetizan las diferentes proteínas las cuales son encargadas de llevar a cabo las diferentes funciones de la célula. De esta forma, cuando los genes se expresan determinan el estado celular y modificando su expresión, provocan un cambio en la célula que puede llevar de un estado sano a uno patológico o viceversa.

En este sentido, una de las técnicas utilizadas para estudiar las variaciones en los niveles de expresión de los genes es el análisis de microarrays. Las microarrays o matrices de expresión génica son, como su propio nombre indica, matrices en las que encontramos diferentes genes frente a diversas condiciones muestrales. De esta manera se analizan los niveles de expresión de los genes para las diferentes condiciones muestrales y se puede determinar en cada momento cuál es el conjunto de genes que se está expresando.

La información generada por las microarrays dependerá de las condiciones experimentales aplicadas. Es decir, dependiendo de las condiciones experimentales, la microarray nos proporciona información de diferente tipo. Por ejemplo si las condiciones experimentales de la microarray son sobre el cáncer de colon, la microarray nos mostrará los niveles de expresión de los genes para el cáncer de colon, si en la microarray se estudia el efecto de fumar tabaco sobre los pulmones, la microarray nos proporcionará el nivel de expresión de los genes del tejido pulmonar bajo estas condiciones. De esta forma las condiciones experimentales pueden proporcionar la respuesta génica a distintos fármacos, a variaciones en las dosis, a diferentes fases en el progreso de una enfermedad, a diferentes tipos de células, a diferentes tipos de tejidos, al género u otras características de los pacientes, etc.

El gran problema de las microarrays es que los datos que proporciona son de tal magnitud (puede haber matrices de expresión génica de 50.000 genes por 5.000 condiciones muestrales) que es difícil abstraer el significado biológico, es decir, analizarlos. Para conseguir este objetivo se han diseñado diferentes aplicaciones (GEO [1], BIOREL [2], ArrayExpress [3], MicroGen [4], etc.) pero ninguna de ellas conduce a una visión holística de lo que sucede en la célula.

PCOPGene-Net [5] es una aplicación web creada por el IBB (Instituto de Biotecnología y Biomedicina) [12] pensada para facilitar el estudio de las relaciones entre las expresiones génicas bajo las condiciones de las microarrays que se analicen. Por medio de la visión global que esta facilita se muestra la red de relaciones entre los genes y mediante la visión en detalle se muestran las variaciones de la relación de expresión entre los genes. Esta y otras aplicaciones para el análisis de microarrays con las que se interrelaciona [5, 7, 8, 9, 10,11] están disponibles en el server del IBB [13].

Las librerías JUNG de Java para la visualización de datos en grafos vía web [6] son utilizadas para montar el gráfico interactivo de la visión global de la aplicación PCOPGene-Net [5]. En el proceso de *layout* que se realiza durante el preproceso de los datos, los genes se ubican en el espacio 2D en función del grado de correlación de cada gen con sus vecinos, agrupando los genes en clusters y facilitando la exhibición del *minimum-spanning path* entre cualquier par de genes de la microarray. Esta disposición de los datos facilita al investigador la navegación a través de la nube de genes de la microarray y le proporciona la información de su interés en cada momento.

El server http://revolutionrsearch.uab.es [13] opera actualmente con microarrays del orden los 1.600 genes. Para estas microarrays, PCOPGene-Net [5] generaría grafos interactivos vía web para mostrar las relaciones entre los genes y operar con ellos. Con tal fin, el layout que se realiza en el preproceso de datos trabaja en dos niveles a la hora de distribuir los genes en el espacio 2D, uno para los genes pertenecientes a cada cluster y otro para los clusters de genes. Esta aproximación por niveles permite una visión global y también en detalle del grafo. Sin embargo, la aplicación no está preparada actualmente para mostrar grafos de 10.000 o 30.000 genes. La agrupación por clusters de 5 a 10 genes se queda claramente pequeña. Un ejemplo de esto es que para una microarray de 30.000 genes tendríamos más de 3.000 clusters a organizar en el espacio 2D, a diferencia de los 160 clusters de una microarray de 1.600 genes. El número de relaciones de expresión entre genes también subiría exponencialmente, lo que dificulta estudiar y detectar visualmente sobre el grafo las aristas que representan relaciones de expresión de diferente tipo. Finalmente, la consola de java tiene una capacidad limitada y no es posible mostrar grafos de tal tamaño, ni permitir interactividad para tal número de genes y de aristas.

No obstante la ciencia no se detiene, y la generación de microarrays de gran tamaño es ya una realidad y lo será más en un futuro muy cercano. Con lo que urge encontrar la forma de visualizar y trabajar con grafos interactivos de tal magnitud.

1.3 Objetivos

Como he comentado anteriormente, el server actual opera con microarrays del orden los 1.600 genes. El objetivo principal de este proyecto consiste entonces en modificar dicha aplicación para poder trabajar con matrices de expresión génica de mayor orden. Esto implicará también adaptar el cálculo del layout para la microarray dada, la criba de los datos resultado, y la interfaz web que muestra los resultados y permite operar con ellos.

Actualmente el layout trabaja con dos niveles organizativos. En un primer nivel organiza en el espacio 2D los genes de cada cluster, y en un segundo nivel organiza, en el mismo espacio dimensional, los clusters de genes.

En mi proyecto, al tener un número más elevado de genes, será necesario aumentar los niveles organizativos del layout. Dicho de otra manera, será necesario pasar de trabajar con un layout de dos niveles a trabajar con uno de tres o incluso de cuatro, dependiendo del número de genes y la correlación entre ellos. Tendré entonces una primera ordenación de los genes de cada cluster, otra de los clusters de genes e incorporé la ordenación por hiperclusters, es decir, clusters de clusters, y por hyperclusters de segundo nivel, es decir, clusters de hyperclusters.

Por otra parte, al aumentar el número de genes, obviamente aumenta el número de relaciones entre genes. Por este motivo también será necesario formular los filtros pertinentes para mostrar siempre el número de relaciones entre genes adecuado al tamaño de cada microarray. Este filtro se utilizará en microarrays de todos los tamaños y la cantidad de información relevante variará de una a otra, no solo en términos cuantitativos sino también cualitativos (microarrays con pocos genes por ejemplo, contendrán información más relevante al tratarse de un cluster de genes considerados clave por los creadores de la microarray).

Con el fin de que no se sature la máquina de java y permita operar más fácilmente con el grafo generado, para microarrays muy grandes se repartirán el total de genes en distintos applets. Llamaremos a

estas divisiones particiones de la microarray. De esta forma, en lugar de tener un applet por cada microarray, como pasa en el caso de microarrays de tamaño reducido, tendremos un applet por cada partición de la microarray. Esta partición podrá estar basada en hyperclusters de segundo o primer nivel, dependiendo del tamaño de la microarray y del grado de correlación entre los genes.

Esta modificación implica que ahora los applets no trabajarán con todos los genes que conforman una microarray, sino que lo harán con un subconjunto de éstos. Por este motivo será necesaria la adaptación del applet y del preproceso que prepara los ficheros para ser leídos por el applet.

De forma detallada, los objetivos de mi proyecto son los siguientes:

- Modificaciones en el preproceso para poder analizar microarrays de muy diferentes tamaños:
    - Conseguir la máxima funcionalidad, entendivilidad y operatividad tanto para microarrays pequeñas (hasta 1.600 genes) como para microarrays grandes hasta 40.000 genes. Con todo el rango de tamaños intermedios.
    - Diseño de nuevas fórmulas dependientes del número de genes para cribar las relaciones de expresión no lineales entre genes. Estas relaciones de expresión no lineales son de vital importancia pues nos muestran los cambios de fenotipos.
    - Diseño e implementación de un algoritmo para la división en clusters, hyperclusters y hyperclusters de segundo orden.
    - Adaptación del layout para que funcione con microarrays de gran orden.
    - Partición del layout, el mínimum spanning tree y demás datos necesarios para el applet. Esta partición es necesaria para que dicho applet opere con las particiones de la microarrya (hyperclusters o hyperclusters de segundo nivel) en lugar de con la totalidad de genes de la microarray.
    - Diseño e implementación de un algoritmo para un último filtrado de relaciones de expresión no lineales por tipología, de forma que las relaciones de expresión más significativas de cada tipo puedan ser observadas y analizadas en el applet.

- Adaptaciones en el applet para poder mostrar microarrays de muy diferentes tamaños:
    - Tratamiento diferenciado de las microarrays pequeñas y de gran tamaño.
    - Para microarrays de gran tamaño, el applet operará solo con parte de los genes de la microarray. Las diferentes particiones de la microarray estarán contenidas en diferentes applets abiertos simultáneamente. La parte de genes de la microarray contenida en cada applet se corresponderá con un hypercluster diferente, de primer o segundo nivel, dependiendo del tamaño de la microarray y del grado de correlación entre los genes.
    - Para los applets que solo operan con parte del total de genes de la microarray, coordinación con las aplicaciones externas al applet y coordinación entre los distintos applets que contienen las diferentes particiones de la microarray.

- Adaptaciones del aplicativo web:
    - El aplicativo web abrirá simultáneamente los diferentes applets que muestran las particiones que conforman el total de genes de la microarray analizada.

.

1.4 Organización de la memoria

Para llevar a cabo los objetivos propuestos anteriormente he seguido la planificación que se expone a continuación.

En una primera fase me dediqué a adquirir los conocimientos necesarios sobre la bioinformática y a familiarizarme con el entorno de las microarrays. También me familiaricé con la aplicación web así como con los diversos métodos para acceder y tratar los datos de microarrays.

Una vez comprendidos los pasos a seguir para el tratamiento de microarrays, en una segunda fase arreglé y mejoré el código que permitía trabajar con matrices de expresión génica de tamaño reducido para conseguir un funcionamiento óptimo del preproceso de datos.

En la tercera fase empecé a trabajar con microarrays de gran tamaño. El objetivo de esta fase era encontrar un método para dividir las microarrays en diversas particiones. Para ello ideé un algoritmo que permitía agrupar los genes de forma dinámica en clusters, hyperclusters o hyperclusters de segundo nivel. Posteriormente me servía de esta agrupación para repartir los genes y crear clusters, mínimum spanning trees, y layouts de forma separada para cada partición de la microarray. Finalmente dividía los datos generados por otros análisis del preproceso para los genes que irían a parar a cada diferente partición.

En la cuarta fase realicé las adaptaciones necesarias al applet para que éste fuera capaz de operar con las diferentes particiones de la microarray. En esta fase fue necesario el estudio de las distintas aplicaciones externas que lanza el applet y la posterior adaptación de algunas de ellas.

Dado el gran volumen de genes con el que trabajé, pude apreciar que el número de relaciones de expresión no lineales entre los genes se disparaba. Por este motivo fue necesaria una quinta fase. En ésta diseñé e implementé un algoritmo para el filtrado de relaciones de expresión no lineales entre los diferentes genes. De esta forma obtuvimos dos grandes ventajas; por un lado una visualización más enfocada y clara de la información y por otro, descartamos posibles problemas posteriores debidos a la sobrecarga de datos.

Finalmente, la última fase del proyecto consistió en la adaptación del aplicativo web para conseguir mostrar simultáneamente todas las particiones que conforman una microarray.

Cabe destacar que durante cada fase del trabajo he realizado diferentes modificaciones para mejorar la funcionalidad, operatividad y usabilidad del conjunto de la aplicación con la supervisión y asesoramiento del codirector Mario Huerta.

En los capítulos venideros se describe con minuciosidad el trabajo realizado. La estructura seguida es la siguiente:

- Fundamentos teóricos: Expongo los conocimientos biológicos y técnicos necesarios para entender el proyecto.
- Fases: Describo el trabajo desarrollado para resolver el proyecto, los problemas encontrados y la solución adoptada.
- Resultados: Detallo los resultados obtenidos con la elaboración de mi proyecto.
- Informe técnico: Describo los programas implementados y la estructura de directorios detalladamente para facilitar su reusabilidad y adaptabilidad.
- Conclusiones
- Bibliografía

2. Fundamentos teóricos

2.1 Bioinformática

La Bioinformática es una disciplina científica emergente que utiliza tecnología de la información para organizar, analizar y distribuir información biológica con la finalidad de responder preguntas complejas en biología. Bioinformática es un área de investigación multidisciplinaria, la cual puede ser ampliamente definida como la interfase entre dos ciencias: Biología y Computación y está impulsada por la incógnita del genoma humano y la promesa de una nueva era en la cual la investigación genómica puede ayudar dramáticamente a mejorar la condición y calidad de vida humana.

Avances en la detección y tratamiento de enfermedades y la producción de alimentos genéticamente modificados son entre otros ejemplos de los beneficios mencionados más frecuentemente. Involucra la solución de problemas complejos usando herramientas de sistemas y computación. También incluye la colección, organización, almacenamiento y recuperación de la información biológica que se encuentra en base de datos.

Según la definición del Centro Nacional para la Información Biotecnológica "National Center for Biotechnology Information" (NCBI por sus siglas en Inglés, 2001): "Bioinformática es un campo de la ciencia en el cual confluyen varias disciplinas tales como: biología, computación y tecnología de la información. El fin último de este campo es facilitar el descubrimiento de nuevas ideas biológicas así como crear perspectivas globales a partir de las cuales se puedan discernir principios unificadores en biología. Al comienzo de la "revolución genómica", el concepto de bioinformática se refería sólo a la creación y mantenimiento de base de datos donde se almacena información biológica, tales como secuencias de nucleótidos y aminoácidos. El desarrollo de este tipo de base de datos no solamente significaba el diseño de la misma sino también el desarrollo de interfaces complejas donde los investigadores pudieran acceder los datos existentes y suministrar o revisarlos.

Posteriormente toda esa información debía ser combinada para formar una idea lógica de las actividades celulares normales, de tal manera que los investigadores pudieran estudiar cómo estas actividades se veían alteradas en estados de una enfermedad. De allí viene el surgimiento del campo de la bioinformática y ahora el campo más popular es el análisis e interpretación de varios tipos de datos, incluyendo secuencias de nucleótidos y aminoácidos, dominios de proteínas y estructura de proteínas.

El proceso de analizar e interpretar los datos es conocido como biocomputación. Dentro de la bioinformática y la biocomputación existen otras sub-disciplinas importantes:

- El desarrollo e implementación de herramientas que permitan el acceso, uso y manejo de varios tipos de información.
- El desarrollo de nuevos algoritmos (fórmulas matemáticas) y estadísticos con los cuales se puedan relacionar partes de un conjunto enorme de datos, como por ejemplo métodos para localizar un gen dentro de una secuencia, predecir la estructura o la función de proteínas y poder agrupar secuencias de proteínas en familias relacionadas.

La Medicina Molecular y la Biotecnología constituyen dos áreas prioritarias científico tecnológicas como desarrollo e Innovación Tecnológica. El desarrollo en ambas áreas está estrechamente relacionado. En ambas se pretende potenciar la investigación genómica y postgenómica así como de la bioinformática, he-

rramienta imprescindible para el desarrollo de estas. Debido al extraordinario avance de la genética molecular y la genómica, la Medicina Molecular se constituye como arma estratégica del bienestar social del futuro inmediato. Se pretende potenciar la aplicación de las nuevas tecnologías y de los avances genéticos para el beneficio de la salud. Dentro de las actividades financiables, existen acciones estratégicas, de infraestructura, centros de competencia y grandes instalaciones científicas. En esta área, la dotación de infraestructura se plasmará en la creación y dotación de unidades de referencia tecnológica y centros de suministro común, como Centros de Bioinformática, que cubran las necesidades de la investigación en Medicina Molecular. En cuanto a centros de competencia, se crearán centros de investigación de excelencia en hospitales en los que se acercará la investigación básica a la clínica, así como centros distribuidos en red para el apoyo a la secuenciación, DNA microarrays y DNA chips, bioinformática, en coordinación con la red de centros de investigación genómica y proteómica que se proponen en el área de Biotecnología. En esta área la genómica y proteómica se fundamenta como acción estratégica o instrumento básico de focalización de las actuaciones futuras.

Las tecnologías de la información jugarán un papel fundamental en la aplicación de los desarrollos tecnológicos en el campo de la genética. La aplicación de los conocimientos en genética molecular y las nuevas tecnologías son necesarias para el mantenimiento de la competitividad del sistema sanitario no sólo paliativo sino preventivo. La identificación de las causas moleculares de las enfermedades junto con el desarrollo de la industria biotecnológica en general y de la farmacéutica en particular permitirán el desarrollo de mejores métodos de diagnóstico, la identificación de dianas terapéuticas y desarrollo de fármacos personalizados y una mejor medicina preventiva.

2.2 Microarrays

Un chip de ADN (del inglés DNA microarrays) es una superficie sólida a la cual se unen una serie de fragmentos de ADN. Las superficies empleadas para fijar el ADN son muy variables y pueden ser vidrio, plástico e incluso chips de silicio. Los arreglos de ADN son utilizados para averiguar la expresión de genes, monitorizándose los niveles de miles de ellos de forma simultánea.

La técnica consiste en extraer el RNAm de una célula buena y de otra experimental mediante isolation RNA. Una vez extraídos los dos RNAm se marca cada uno de ellos con un color distinto y se combinan los dos. Acto seguido se vierte el combinado en la superficie del chip de tal modo que cada RNAm se unirá o no a los cDNA de cada gen del chip. Finalmente, aplicando técnicas de análisis de imágenes es posible generar una matriz de datos numéricos, a partir de los patrones de intensidades de cada celda y discriminando la señal informativa de ruido que pudiera haber en segundo plano. Estos datos numéricos corresponderán al nivel de expresión de cada gen expuesto a una serie de condiciones muestrales. Por ejemplo, si el ARN de la célula experimental se coloreó de color rojo, los genes con un color más cercano al rojo tienen un nivel de expresión más elevado en las células experimentales.

Actualmente existen diferentes bases de datos a nuestro alcance a través de internet que unifican y facilitan toda esta información genética además de ofrecer diversas herramientas para el análisis de esta gran cantidad de información. Algunas de estas bases de datos por ejemplo son las que hay en el EMBL (European Molecular Biology Laboratory), el SIB (Swiss Institute of Bioinformatics), el EBI (European Bioinformatics Institute) o el NCBI (National Center for Biotechnology Information). El EBI y el NCB son los que más información contienen y por lo tantos los más utilizados.

Por lo tanto las microarrays son una potente fuente de obtención de perfiles de expresión de genes sometidos a diferentes condiciones, identificar los patrones de los niveles de expresión será muy útil para compararlos y poder estudiar las respuestas de los genes.

Aplicando una serie de procesos experimentales y computacionales sobre la microarray se obtiene una matriz numérica bidimensional que consta de los genes de poblaciones distintas como individuos y de las condiciones muestrales a las que se expusieron las células como variables en el caso que se quiera estudiar a los genes, o a la inversa, si es que se quiere realizar un estudio comparativo de las condiciones a que se somete. Cada uno de los valores de la matriz representa el nivel de expresión de un determinado gen bajo una cierta condición muestral. Es posible que en algunos casos se produzcan errores en el proceso y se generen huecos, estos huecos tiene valor 0 dentro de la matriz.

Pueden existir microarrays de muchos y diversos tamaños pero normalmente no suelen tener menos de 500 genes por 60 condiciones muestrales.

Dado que realizar una análisis directo de estas matrices es una tarea prácticamente imposible se hacen necesarias técnicas computacionales que permitan agrupar todos estos datos y a partir del agrupamiento realizar el análisis biológico. La técnica más utilizada es el agrupamiento o clustering.

2.3 Clustering

El objetivo de los algoritmos de clustering es, dada una matriz de individuos y variables, encontrar un grupo (clúster) de un conjunto de individuos, de tal forma que los clusters resultantes sean homogéneos y/o estén bien separados.

El punto clave es reducir la gran cantidad de datos caracterizándolos en grupos más pequeños de individuos similares. Esto implica que los individuos pertenecientes a un mismo clúster son lo más similares posibles entre ellos, mientras que los individuos de clusters distintos son lo más disimilares posibles.

La agrupación de los individuos se realizará de acuerdo a la separación entre ellos determinada por una medida de distancia dada, llamada medida de disimilaridad. Este tipo de clustering es estadístico. En cambio, cuando la agrupación de los individuos se realiza de acuerdo a un criterio biomédico por parte del investigador, es un tipo de clustering por criterios biomédicos.

2.4 Hierarchical clustering

El hierarchical clustering o agrupación jerárquica, es un método de análisis de clusters que busca construir una jerarquía de clusters. Generalmente, las estrategias de hierarchical clustering se dividen en dos tipos: Aglomerativa o Divisiva.

Los resultados de la hierarchical clustering se suelen presentar en un dendrograma.

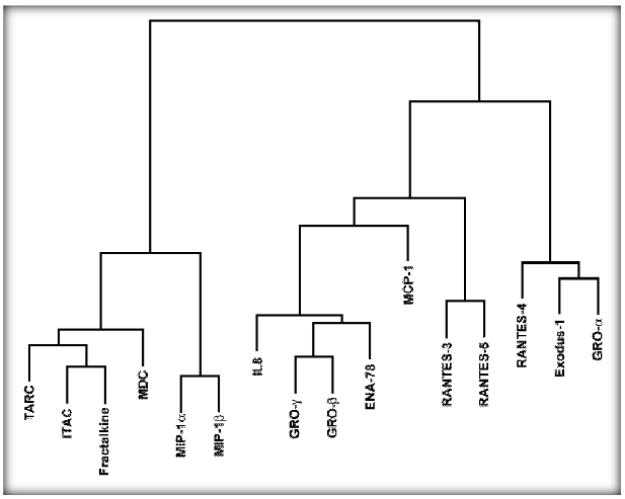


Figura 2-1: Ejemplo de un hierarchical clustering de tipo aglomerativo. Los genes, representados a los pies de la imagen se van uniendo por correlación a medida que se asciende de nivel.

2.5 Minimum Spanning Tree

Dado un grafo conexo, un mínimum spanning tree (MST) o en español, árbol recubridor mínimo, de ese grafo es un subgrafo que tiene que ser un árbol y contener todos los vértices del grafo inicial. Cada arista tiene asignado un peso proporcional entre ellos, que es un número representativo de algún objeto, distancia, etc., y se usa para asignar un peso total al árbol recubridor mínimo computando la suma de todos los pesos de las aristas del árbol en cuestión.

Un minimun spanning tree es un árbol recubridor que pesa menos o igual que otros árboles recubridores. Todo grafo tiene un bosque recubridor mínimo.

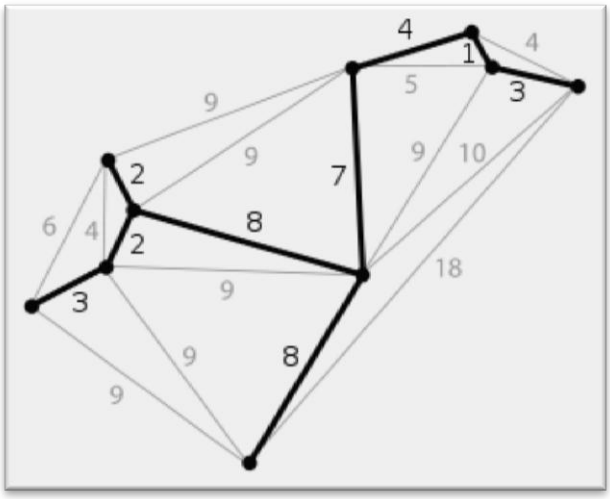


Figura 2-2: Un ejemplo de mínimum spanning tree. Cada punto representa un vértice, cada arista está etiquetada con su peso, que en este caso equivale a su longitud.

Un ejemplo sería una compañía de cable trazando cable a una nueva vecindad. Si está limitada a trazar el cable por ciertos caminos, entonces se hará un grafo que represente los puntos conectados por esos caminos. Algunos de estos caminos podrán ser más caros que otros, por ser más largos. Estos caminos serían representados por las aristas con mayores pesos. Un árbol recubridor para este grafo sería un subconjunto de estos caminos que no tenga ciclos pero que mantenga conectadas todas las casas. Puede haber más de un árbol recubridor posible. El MST será el de menos coste.

En el caso de un empate, porque podría haber más de un MST; en particular, si todos los pesos son iguales, todo árbol recubridor será mínimo. De todas formas, si cada arista tiene un peso distinto existirá sólo un MST. La demostración de esto es trivial y se puede hacer por inducción. Esto ocurre en muchas situaciones de la realidad, como con la compañía de cable en el ejemplo anterior, donde es extraño que dos caminos tengan exactamente el mismo coste. Esto también se generaliza para los bosques recubridores.

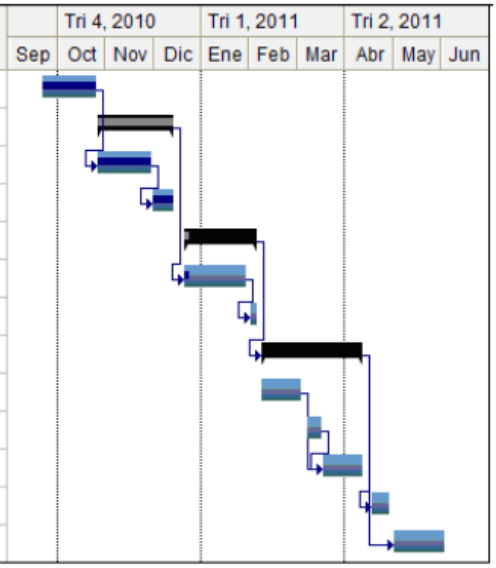
Si los pesos son positivos, el MST es el subgrafo de menor costo posible conectando todos los vértices, ya que los subgrafos que contienen ciclos necesariamente tienen más peso total.

3. Fases

3.1 Comparativa entre las planificaciones inicial y final

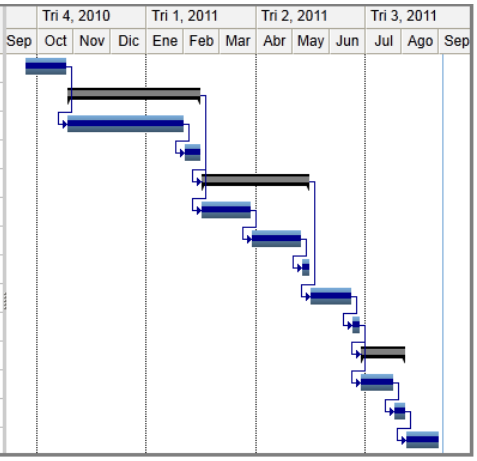
A continuación expongo la planificación inicial que había planteado para el desarrollo de este proyecto:

	Nombre	Duración	Inicio	Fin
✓	1.- Introducción	25d?	21/09/2010	25/10/2010
✓	**2.- Pruebas de código**	**35d?**	**26/10/2010**	**13/12/2010**
✓	2.1.- Corrección del código de partida	25d?	26/10/2010	29/11/2010
✓	2.2.- Comparación de microarrays	10d?	30/11/2010	13/12/2010
	3.- Modificación del algoritmo del layout	**35d?**	**20/12/2010**	**04/02/2011**
	3.1.- Realización del layout a 3 niveles	30d?	20/12/2010	28/01/2011
	3.2.- Comprobación	5d?	31/01/2011	04/02/2011
	4.- Interfaz web	**47d?**	**07/02/2011**	**12/04/2011**
	4.1.- Creación del applet de hiperclusters	20d?	07/02/2011	04/03/2011
	4.2.-Adaptación de applets	8d?	08/03/2011	17/03/2011
	4.3.- Comunicación entre applets	18d?	18/03/2011	12/04/2011
	5.- Optimización	10d?	18/04/2011	29/04/2011
	6.- Redacción de la memória	25d?	02/05/2011	03/06/2011



Seguidamente, adjunto la planificación real del proyecto y explico los motivos por los cuales las dos planificaciones son distintas.

		Nombre	Duración	Inicio	Fin
1	✓	1.- Introducción	25d?	21/09/2010	25/10/2010
2	✓	**2.- Pruebas de código**	80d?	**26/10/2010**	**14/02/2011**
3	✓	2.1.- Corrección del código de partida	70d?	26/10/2010	31/01/2011
4	✓	2.2.- Comparación de microarrays	10d?	01/02/2011	14/02/2011
5	✓	**3.- Modificación del algoritmo del layout**	65d?	**15/02/2011**	**16/05/2011**
6	✓	3.1.- División en particiones de microarray	30d?	15/02/2011	28/03/2011
7	✓	3.2.-Modificación de las fórmulas para el filtrado de curvas	30d?	29/03/2011	09/05/2011
8	✓	3.2.- Comprobación	5d?	10/05/2011	16/05/2011
9	✓	4.- Modificación del applet	25d?	17/05/2011	20/06/2011
10	✓	5.- Filtrado de aristas no lineales	5d?	21/06/2011	27/06/2011
11	✓	**6.- Interfaz web**	28d?	**28/06/2011**	**04/08/2011**
12	✓	6.1.- Creación del applet de hiperclusters	20d?	28/06/2011	25/07/2011
13	✓	6.2.-Adaptación de applets	8d?	26/07/2011	04/08/2011
14	✓	7.- Redacción de la memória	20d?	05/08/2011	01/09/2011



Si observamos esta planificación con detenimiento, podemos ver que la fase que más ha cambiado respecto a la planificación inicial ha sido la que engloba la corrección del código de partida. Esto es debido a los diferentes problemas que he ido encontrando, los cuales he solucionado con éxito, pero después de concienzudos análisis, búsquedas de alternativas y corrección de diversos errores.

No obstante, el principal motivo de retraso ha sido la propia naturaleza de los datos a analizar. Al no haberse trabajado nunca con microarrays tan grandes, era imposible de prever que los datos fueran a estar tan correlacionados como lo están realmente. Algo que solo es posible detectar una vez los datos ya han

sido analizados. Este hecho ha implicado el diseño e implementación de diversos procesos de filtrado con los que no se contaba inicialmente, pero que son totalmente necesarios para obtener un resultado óptimo.

Por otra parte, la idea que se tenía era dividir directamente las microarrays mediante sus hyperclusters. No obstante, al existir genes tan correlacionados tuve que realizar el diseño e implementación del proceso de partición de forma que las particiones se pudieran realizar con clusters, hyperclusters o hyperclusters de segundo nivel dependiendo del grado de correlación de los genes. De esta forma se consigue que las particiones de la microarray sean óptimas.

3.2 Conocimientos previos en el ámbito de la bioinformática y del proyecto

Antes de empezar a realizar mi proyecto, tuve que adquirir ciertos conocimientos sobre la bioinformática dado que anteriormente no había tenido la posibilidad de adentrarme en este campo. También fue necesario familiarizarme y comprender el entorno de la aplicación web así como con los diversos métodos para acceder y tratar la información almacenada.

3.2.1 Adquirir conocimientos sobre la bioinformática

Para poder realizar correctamente el trabajo es necesario aprender los conceptos fundamentales con la intención de comprender en su plenitud el proyecto escogido. Los conceptos adquiridos fueron en referencia al análisis de microarrays, donde se engloba la aplicación web implementada. Los conceptos propiamente dichos se muestran en la anterior sección de fundamentos teóricos. En ésta se hace especial hincapié al concepto de una microarray, es decir, una matriz en la que encontramos diferentes genes frente a diversas condiciones muestrales.

Otro concepto indispensable para el desarrollo de este trabajo es el concepto de clúster. Tal y como he comentado anteriormente, un clúster es la agrupación de genes pertenecientes a una microarray por tener un comportamiento parecido o similar, ya sea estadístico o por criterios biomédicos.

3.2.2 Familiarizarme con el aplicativo PCOPGene-Net

Comprender y familiarizarme con el aplicativo PCOPGene-Net [5] es importante ya que es el lugar donde se integra mi aplicación.

La herramienta PCOPGene [5] es una aplicación web para el estudio de microarrays. Hay dos vertientes para su utilización. En la primera vía, a partir de un gráfico de genes marcadores permite seleccionarlos con la intención de poder estudiarlos. Ofrece también la posibilidad de relacionar los genes con publicaciones donde aparece el gen, permite mostrar información relevante sobre el gen, etc. En la segunda vía, tiene una serie de herramientas que permiten analizar la microarray y su comportamiento, la relación entre los diferentes clústers y cómo se comportan, etc.

A continuación, en la figura 3-1 se puede observar la interfaz y se detallan las diversas opciones que brinda esta herramienta:

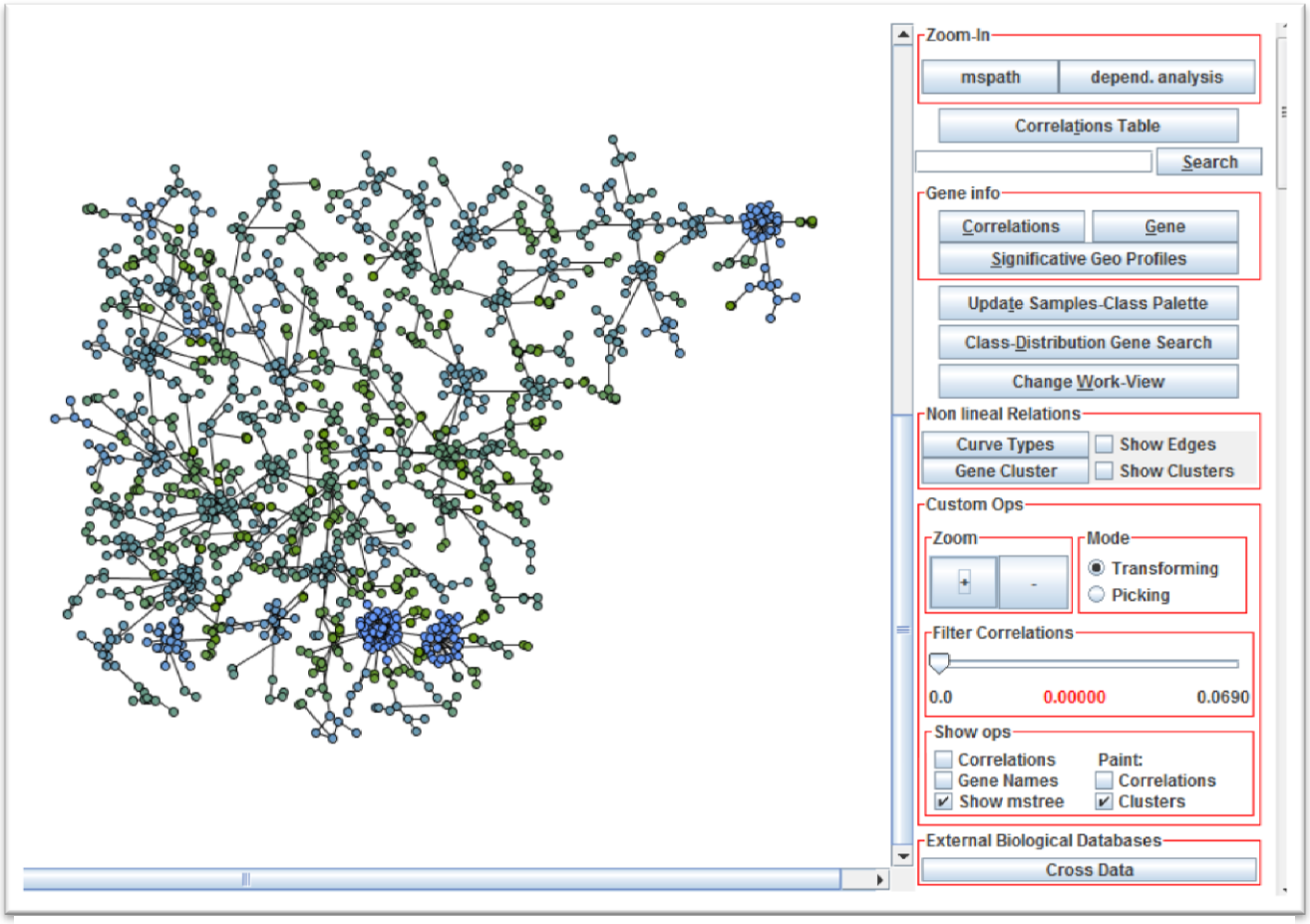


Figura 3-1: Interfaz de la aplicación PCOP Gene. Como se puede observar, esta aplicación permite a los investigadores ver todos los genes que compone su microarray distribuidos en un espacio 2D por la correlación entre sus expresiones (layout) y realizar diferentes operaciones sobre ellos.

- Zoom-In: En un momento dado es posible que el investigador quiera centrar su investigación en un proceso concreto para saber qué genes están involucrados en él y estudiar sus dependencias entre expresiones en detalle. Esta tarea es proporcionada por la operación zoom-in.
    - Mspath: Como su nombre indica, muestra el mspath o camino que sigue el minimun spaning tree para unir los genes seleccionados.
    - Depence analysis: Las fluctuaciones de la dependencia de expresión entre el conjunto de genes suministrados también pueden ser analizadas. En esta aplicación se muestran estas fluctuaciones en una gráfica, en la cual el eje de ordenadas indica el nivel de expresión y el de abscisas indica el parámetro de la relación. Las líneas representan el nivel de expresión de los genes en comparación.
- Correlations Table: Muestra la tabla de correlaciones todos con todos, de los genes que conforman la microarray.
- Search: Introduciendo el nombre de un gen, permite encontrarlo en el grafo.
- Gene Info: Proporciona información sobre los genes seleccionados.

    - Correlations: Muestra las correlaciones que los genes seleccionados mantienen con el resto de genes de la microarray.
    - Gene: Busca los genes seleccionados en la base de datos Gene [14].
    - Significative Geo Profiles: Busca los genes seleccionados en la base de datos Significative Geo Profiles [14].
- Update Sample Class Palette: El usuario elige una distribución en clusters para las condiciones muestrales sobre la que trabajar.
- Class-Distribution Gene Search Esta aplicación tiene la función de encontrar los genes marcadores de la microarray del usuario para la distribución en clusters del mismo y para las condiciones muestrales, a partir de sobreexpresar o infoexpresar unos clusters respecto a otros.
- Change Work Directori: Cambia el directorio de trabajo.
- Non Linear Relations:
    - Curve Types: Muestra una nueva ventana que permite seleccionar los tipos de relaciones de expresión no lineales que mantienen los genes. Seleccionando la opción Show Edges, dichas relaciones se hacen visibles en el grafo.
    - Gene Cluster: Muestra una nueva ventana que permite seleccionar los tipo de clusters de genes basados en las relaciones de expresión no lineales de cada gen. Seleccionando la opción Show Clusters, dichos clusters se hacen visibles en el grafo.
- Custom Ops: Aquí se encuentran diversas opciones que permiten cambiar varios atributos del grafo. A continuación se detallan las posibles opciones:
    - Zoom: Permite alejar o acercar la microarray para tener una visión más detallada de las relaciones entre los genes.
    - Mode: Permite cambiar el modo de actuación del ratón. Si se selecciona *Transforming* el ratón mueve la microarray mientras que en caso contrario, *Picking* permite seleccionar los genes que se quieren estudiar con más detalle,
    - Filter Correlations: Este filtro permite mostrar u ocultar las relaciones entre los diversos genes basándose en la correlación que mantienen.
    - Show ops:
        - Correlations: Muestra en cifras el valor de las correlaciones que mantienen los genes.
        - Gene Names: Muestra, al lado de cada gen, su nombre.
        - Show mstree: Permite visualizar o no el minimun spanning tree de la microarray.
        - Paint:
            - Correlations: Opción que muestra mediante diferentes tonalidades las correlaciones de todos los genes. Las mejores correlaciones aparecen más oscuras mientras que las peores aparecen en tonos más claros.
            - Clusters: Esta opción muestra todos los clusters. Cuando esta activada, como es el caso de la figura 3-1, se muestran todos los genes de un cluster de un color. Si, por el contrario está desactivada, todos los genes aparecen con la misma tonalidad.
- External Biological Databases:
    - Cross Data: Esta opción permite buscar información de los genes seleccionados en diversas bases de datos, artículos, etc.

3.2.2 Familiarizarme con el preproceso para analizar los datos de microarrays pequeñas

El preproceso de los datos de la microarray es el encargado de generar todos los ficheros de resultados que necesitará el aplicativo web PCOPGene [5] para operar: Layout de los genes por correlación, clus-

ters de los genes por correlación, cluster de los genes por el tipo de relaciones que mantienen estos genes con el resto, clasificación de las relaciones de expresión no lineales, etc…

Este preproceso constaba de los siguientes pasos:

1. Cálculo de las correlaciones entre genes y determinación de las relaciones de expresión no lineales entre genes.
2. Búsqueda de los genes mejor correlacionados
3. Búsqueda del gen mejor correlacionado con cada gen
4. Cálculo del minimun spannig tree entre los genes de la microarray
5. Proceso de clustering de genes por la correlación entre sus expresiones
6. Cálculo del Layout de la microarray

Tras un estudio en detalle de cada uno de los pasos anteriores conseguí detectar y solventar ciertas irregularidades que presentaban algunos procesos y que lo hacían imposible de aplicar a grandes microarrays.

3.3 Mejora del preproceso para analizar los datos de microarrays pequeñas

Una vez comprendidos los pasos a seguir para el tratamiento de microarrays, en una segunda fase arreglé y mejoré el código del preproceso que permitía trabajar con matrices de expresión génica de tamaño reducido para conseguir un funcionamiento óptimo del preproceso de datos.

3.3.1 Optimización del cálculo de correlaciones entre genes

Este proceso tiene dos finalidades básicas. Por una parte, dada la microarray en cuestión calcula la correlación que mantiene cada gen con los restantes. Por otra, proporciona toda la información necesaria sobre las relaciones de expresión no lineales que mantienen los genes.

De esta forma, dada una microarray de n genes el proceso de cálculo de correlaciones entre genes permite obtener por una parte un fichero como el que se muestra en la figura 3-2, en el que constan las parejas gen_x - gen_y y la correlación que mantienen.

```
1   2   0.012246
1   3   0.019694
1   4   0.005982
⋮
1   n   0.059434
2   3   0.035903
2   4   0.036961
⋮
n-1 n   0.25541
```

Figura 3-2: Ejemplo de fichero de salida que proporciona el proceso de cálculo de correlaciones. Como se puede observar, en este fichero aparecen las correlaciones de todos los genes con todos los genes.

Tras el estudio de éste proceso y la realización de diversas pruebas, pude comprobar la existencia de casos en los cuales la información obtenida era errónea e incompleta.

En caso que a la microarray le faltase la respuesta de algún gen a la última condición muestral, éste proceso omitía dicho gen y el siguiente y reenumeraba los genes restantes. De esta forma, ante una microarray de 1.400 genes, podía darse el caso de que en el fichero resultante aparecieran solo los genes comprendidos entre el 1 y el 1.300. Para este ejemplo había 50 genes que omitían su respuesta a la última condición muestral.

Con el fin de solucionar esta problemática hice que en los casos en los que faltase la respuesta de un gen a la última condición muestral sólo se omitiera la condición muestral faltante, en lugar del gen en cuestión y su sucesor inmediato. De esta forma, el proceso podía encontrar relaciones de dicho gen con los genes restantes de la microarray y continuar el proceso.

Por otra parte, el proceso de cálculo de correlaciones entre genes también proporciona información relativa a las relaciones de expresión no lineales que mantienen los genes. Dado que este proceso era correcto para microarrays reducidas, en esta fase no fue modificado.

3.3.2 Adaptaciones en la búsqueda de los genes mejor correlacionados

En el proceso de búsqueda de los genes mejor correlacionados se crea un fichero para cada gen de la microarray en el que figuran ordenados por correlación, los 500 genes mejor correlacionados con el primero junto con las correlaciones que mantienen. Dicho de otra manera, para cada gen, se obtiene un fichero en el constan sus 500 mejores relaciones con los genes restantes.

La única modificación necesaria en el proceso de búsqueda de las mejores correlaciones de cada gen fue la eliminación de un parámetro que regulaba el número máximo de genes.

3.3.3 Mejoras en la búsqueda del gen mejor correlacionado con cada gen

El proceso de búsqueda del gen mejor correlacionado con cada gen crea un único fichero en el que constan los diferentes genes junto con su mejor correlación. Es decir, para cada gen x se obtiene la tripleta *x-y-correlación*, donde *y* es el gen que mejor correlación mantiene con el primero.

El problema del proceso de búsqueda de las mejores correlaciones entre los genes de la microarray es que estaba mal diseñado; podía tardar varios minutos en obtener los resultados para una microarray de 1.400 genes. Esto presentaba un gran inconveniente ya que para microarrays de gran tamaño tardaba diversos días, tiempo que no se podía gastar en un cálculo tan simple.

Por este motivo rediseñé el algoritmo de nuevo haciéndolo mucho más óptimo. En lugar de realizar un acceso a fichero por cada gen, como hacía el algoritmo anterior, decidí realizar un único acceso. De esta forma conseguí reducir el tiempo de espera a tan solo milisegundos.

3.3.4 Adaptaciones en el cálculo del mínimum spanning tree entre los genes de la microarray

Este proceso es el encargado de crear el minimun spaning tree (MST). Como he comentado anteriormente, éste es un caso especial de árboles en los que el sumatorio de los pesos de todas las aristas es el mínimo posible.

Gracias al MST es posible obtener una representación donde las aristas de los genes que tienen una correlación más baja entre sí quedará y las de los más alejados no.

Tras estudiar el proceso de cálculo del mínimum spanning tree entre los genes de la microarray pude comprobar que efectivamente estaba implementado de forma óptima, motivo por el cual la única modificación necesaria fue la ampliación de un parámetro que regulaba el número máximo de genes con el que podía trabajar.

3.3.5 Proceso de clustering de genes por la correlación entre sus expresiones

Una vez obtenidos todos los genes con sus correlaciones y el MST (minimun spaning tree), el siguiente paso es agrupar los genes de manera que los que estén más correlacionados entre sí pertenezcan a un mismo cluster.

Como he comentado anteriormente, la ventaja de tener clusters de genes por correlación es que es posible identificar aquellos genes que mantienen una alta dependencia entre sus expresiones y situarlos así en la interfície gráfica de forma más cohesionada. A diferencia del MST, que también relaciona los genes entre ellos por la correlación de sus expresiones, el objetivo aquí es identificar sobre el árbol, los genes que podrían ser considerados del mismo cluster por mantener una alta correlación entre todos ellos.

Para llevar a cabo esta tarea se utiliza la técnica de análisis de clusters, consistente en métodos utilizados para dividir objetos en clusters de forma que el grado de similitud/asociación entre miembros de un mismo cluster sea más fuerte que el grado de similitud/asociación entre miembros de diferentes clusters.

Este tipo de análisis es un método que permite descubrir asociaciones y estructuras en los datos que muchas veces no son evidentes a priori pero que pueden ser útiles una vez halladas.

Para calcular los clusters de genes se utilizan los resultados obtenidos en el paso 3 (Búsqueda del gen mejor correlacionado con cada gen), es decir la mejor correlación que mantiene cada uno de los genes.

De esta forma, para hallar los clusters de genes se siguen estos pasos:

1. Obtener una tabla en la que figuren todos los genes junto con el gen con el que mantienen una mayor correlación.
2. Recorrer la tabla anterior y estudiar en cada caso el gen asociado.
   2.1 Si el gen asociado se encuentra ya en un cluster se añade el gen inicial al mismo cluster.
   2.2 En caso contrario se crea un nuevo cluster con los dos genes.
3. Tanto en el caso 2.1 como en el 2.2 es necesario mirar si el gen inicial se encuentra ya en un cluster y, en este caso, si se encuentra en el mismo cluster que el gen asociado. En caso contrario los dos clusters serán fusionados.

Para acabar de comprender el proceso de clustering, a continuación detallo un ejemplo en el que se encuentran las tres posibles situaciones que se pueden dar:

	Tabla: gen – gen mejor correlacionado				
1 →	1	3		6	9
	2	8		7	9
	3	1		8	2
2 →	4	8		9	6
3 →	5	7		10	3

← 4
← 5
← 6

1. El gen 1 no se encuentra en ningún cluster y su gen relacionado (el 3) tampoco. Ponemos los dos genes en el primer cluster libre, en este caso el 1.

Tabla: cluster - genes	
1	1, 3

2. En este punto la tabla cluster – genes contiene la siguiente información.

Tabla: cluster - genes	
1	1, 3
2	2, 8

3. El gen 8 ya está en el segundo cluster y, dado que el gen 4 no está colocado, lo ponemos directamente en el mismo grupo.

Tabla: cluster - genes	
1	1, 3
2	2, 8, 4

4. En este punto la tabla cluster – genes contiene la siguiente información.

Tabla: cluster - genes	
1	1, 3
2	2, 8, 4
3	5, 7
4	6, 9

5. El gen 9 ya está en un cluster pero el 7 también. Tal y como se ha explicado anteriormente fusionaremos ambos clusters en uno solo, el 3.

Tabla: cluster - genes	
1	1, 3
2	2, 8, 4
3	5, 7, 6, 9

6. Finalmente, la tabla cluster-genes resultante es:

Tabla: cluster - genes	
1	1, 3, 10
2	2, 8, 4
3	5, 7, 6, 9

Para finalizar el proceso de clustering es necesario incluir una nueva funcionalidad que es imprescindible de cada a posteriores operaciones.

Esta funcionalidad añadida consiste en detectar todas las aristas intercluster, es decir, aquellas aristas que forman parte del MST pero que pertenecen a dos genes de diferentes clusters. Estas aristas intercluster nos permitirán relacionar los clusters entre sí, unir los clusters con mejor correlación y construir hyperclusters. El concepto de arista intercluster se puede ver de forma más clara en la figura 3-3.

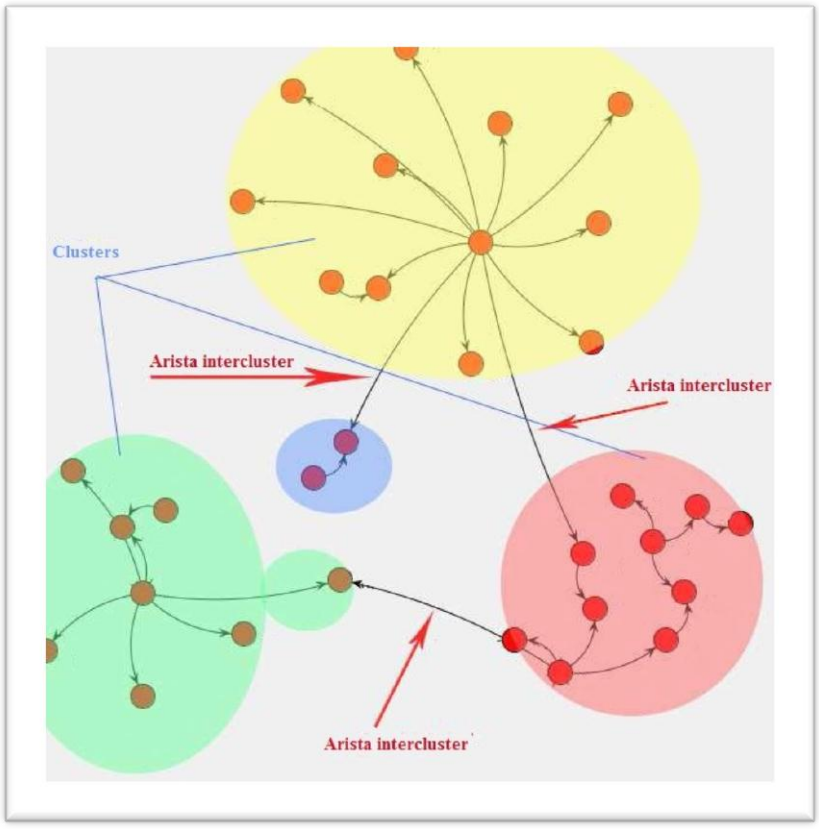


Figura 3-3: Representación de las aristas intercluster. En esta figura se pueden observar los genes (nodos del grafo) englobados en los clusters de genes (pintados en diversos colores) y las tres aristas intercluster que los relacionan.

3.3.6 Optimizaciones en el cálculo del layout

El programa que realiza el layout tiene como objetivo generar las coordenadas de cada gen en función de la correlación entre los genes de la microarray.

Para calcular estas coordenadas se basa en las correlaciones que mantienen los genes de cada cluster entre ellos. Toma éstas correlaciones como distancias y calcula el layout local, es decir, el layout de los genes dentro de sus clusters.

Posteriormente, basándose en las correlaciones de las aristas intercluster crea el layout global, es decir, calcula la posición de los clusters en el espacio 2D y reorienta adecuadamente los genes dentro de cada cluster respetando el layout local del cluster.

Finalmente asigna un color a todos los genes pertenecientes a un mismo cluster basado en la correlación entre los miembros del cluster.

Tras estudiar el proceso de cálculo del layout pude observar que había casos que conducían a error. Un ejemplo de ello es el caso de que un gen esté en el centro del cluster al que pertenece. El poder testear el programa con microarrays de gran tamaño ha facilitado ajustarlo para que funcione en absolutamente todos los casos.

3.4 Tratamiento de microarrays de gran tamaño

En la tercera fase empecé a trabajar con microarrays de gran tamaño. El objetivo de esta fase fue encontrar un método para dividir las microarrays en diversas particiones. Para ello ideé un algoritmo que permitía agrupar los genes de forma dinámica en clusters (nivel 0), hyperclusters (clusters de nivel 1) o hyperclusters de segundo nivel (clusters de nivel 2). Posteriormente me servía de éstos clusters de nivel n para repartir los genes y crear clusters, mínimum spanning trees, y layouts de forma separada para cada partición de la microarray. Finalmente dividía los datos generados por otros análisis del preproceso para los genes que irían a parar a cada diferente partición.

A continuación explico todo este proceso de forma detallada.

3.4.1 Comprobación del grado de correlación entre los genes

Tras estudiar diversas microarrays de gran tamaño pude comprobar que normalmente sus genes están mucho más correlacionados que los de una microarray de tamaño reducido. Esto es debido a que en la mayoría de microarrays de gran tamaño, existen gran cantidad de genes pertenecientes al mismo proceso biológico y que por lo tanto se expresan simultáneamente. También aparecen genes con rango de expresión sin variabilidad, porque no han sido descartados, con lo que suelen dar altas correlaciones cuando se comparan con genes con altos rangos de expresión. Por el contrario, cuando se trabaja con microarrays de tamaño reducido normalmente se utilizan genes de procesos biológicos diferentes, motivo por el cual sus expresiones no están tan correlacionadas. A su vez son genes expresamente seleccionados y cribados, con lo que todos suelen tener unos rangos de expresión altos.

Por estos motivos, en el estudio de grandes microarrays he podido comprobar que existen correlaciones entre genes menores que $1 \cdot 10^{-6}$, valor mínimo que permite el programa. En estos casos, la correlación entre los genes era informada con 0, hecho que causaba muchos y diversos errores en todo el proceso.

Para solucionar estos casos he incluido dos nuevas funcionalidades. Por una parte la detección y por otra su posterior corrección.

La parte de detección consiste, como su nombre indica en detectar los casos en los que se tiene una correlación tan pequeña. En cuanto se detecta una correlación de éste tipo, se modifica automáticamente por el menor valor aceptado por el programa, es decir, $1 \cdot 10^{-6}$.

La parte de corrección tiene como finalidad corregir los errores que hayan inducido las modificaciones anteriores. Supongamos que un gen x tiene entre otras las siguientes relaciones:

x – 1 – 0.000077
x – 2 – 0.000035
x – 3 – 0.000001
x – 4 – 0.000063
x – 5 – 0.000001

En el fichero de mejores correlaciones aparecerá como su mejor correlación la que mantiene con el gen 3, ya que el proceso de búsqueda del gen mejor correlacionado con cada gen únicamente elige la mejor correlación y en caso de empate se queda con la primera que haya aparecido.

Como se puede observar, este no es un criterio significativo. De ahí la existencia del proceso de corrección. Este proceso investiga estos casos y los corrige basándose en la estructura del MST. Como he comentado anteriormente, el MST se crea buscando que el sumatorio de los pesos de todas las aristas sea el mínimo posible. Un criterio que sí es eficiente.

Así, el proceso de comprobación se basará en corregir los errores producidos por correlaciones del mismo peso seleccionando en estos casos, la arista de menor correlación que figure en el MST.

3.4.2 Proceso de clustering de genes por la correlación entre sus expresiones

Una vez obtenido el fichero que me permite saber las correlaciones que mantienen los genes entre sí (figura 3-2) y el MST que relaciona todos los genes de la microarray en base a la correlación de sus expresiones, se realiza el proceso de clustering. En la figura 3-6 se puede apreciar el algoritmo que sigue dicho proceso.

Cabe destacar que el clustering final se corresponde con un hierarchical clustering con single linkage. De hecho utilizo single linkage porque así el hierarchical clustering coincide con el minimum spanning tree, es decir las aristas intercluster del mininmum spanning tree representarán las mejores correlaciones entre los clusters, indicando qué clusters deberán unirse para formar un hypercluster. Esto es así ya que el single linkage correlaciona los clusters basándose en la mejor correlación existente entre todos los genes de un cluster y todos los genes del otro cluster. Por este motivo esta correlación entre genes, que sirve para correlacionar los clusters y formar un hypercluster, será la arista del minimum spanning tree que cruza de un cluster a otro, puesto que el mínimum spanning tree es el árbol de dispersión mínima y siempre tomará la mejor correlación para trazar su camino.

Por otra parte, y con el fin de encontrar las particiones de la microarray de una manera óptima, en el proceso de clustering no sólo calculo los diferentes clusters e hyperclusters de primer y segundo nivel, sino que además, me interesa saber especialmente el número de clusters de nivel n que obtengo. Dicha información es relevante para que las particiones de la microarray se realicen con los clusters de nivel n óptimos; y es que no es lo mismo unir 200 clusters que 15 hyperclusters, por ejemplo.

En este momento sería lógico pensar que por el motivo anterior se podrían calcular las particiones directamente a partir de los hyperclusters de segundo nivel (clusters de mayor nivel con los que trabaja el programa). No obstante, dado que los genes de las microarrays de gran tamaño están muy correlacionados, tampoco sería posible, en ciertos casos, calcular las particiones de la microarray directamente a partir de sus hyperclusters de segundo nivel. Si lo hiciera así encontraría muy pocos hyperclusters de segundo nivel y con una cantidad muy elevada de genes. Por este motivo introduzco ciertas cotas para el número de clusters de nivel n durante el proceso de clustering.

Así, inicialmente se calculan los clusters de genes mediante las aristas intracluster del MST, tal y como se ha explicado en la sección 3.3.5. Una vez tengo el número de clusters de genes de la microarray a analizar, lo comparo con un valor, por defecto 110. En diversos estudios realizados pude comprobar que con un valor mayor de clusters es preferible utilizar un nivel superior de clustering. Pese a que la elección del valor por defecto como 110 no es dependiente del proceso de clustering, mediante la realización de varias pruebas comprobé que era un valor de corte oportuno. En caso que se obtengan más de 110 clusters el programa obtendrá mejores resultados si calcula las particiones de la microarray a partir de un nivel de clusters mayor (hyperclusters o hyperclusters de segundo nivel).

De este modo, en caso que el número de clusters sea superior a 110, procederé a calcular los hyperclusters. Para construir los hyperclusters es necesario realizar un paso previo consistente en encontrar las aristas intercluster del MST, es decir, aristas pertenecientes al MST que unen genes de diferentes clusters.

En el caso de los hyperclusters, considero los clusters como antes los genes y las relaciones intercluster como antes las mejores correlaciones de cada gen. Este nivel de abstracción permite simplificar el cálculo de hyperclusters ya que utilizo el mismo algoritmo para ambos cálculos.

En las figuras 3-4 y 3-5 se puede apreciar una representación esquemática de lo que serían entradas y salidas del proceso de clustering

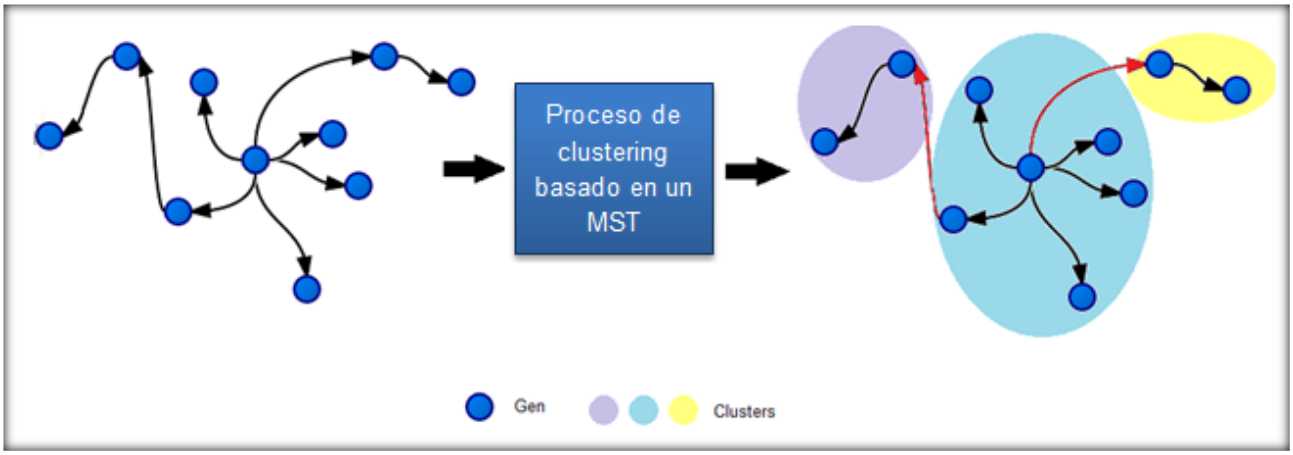


Figura 3-4: Proceso de clustering basado en un MST que toma como entrada los genes junto con las correlaciones entre ellos y como salida proporciona clusters de genes. En la salida se pueden observar los tres clusters obtenidos junto con las aristas intercluster proporcionadas por el MST, marcadas en rojo.

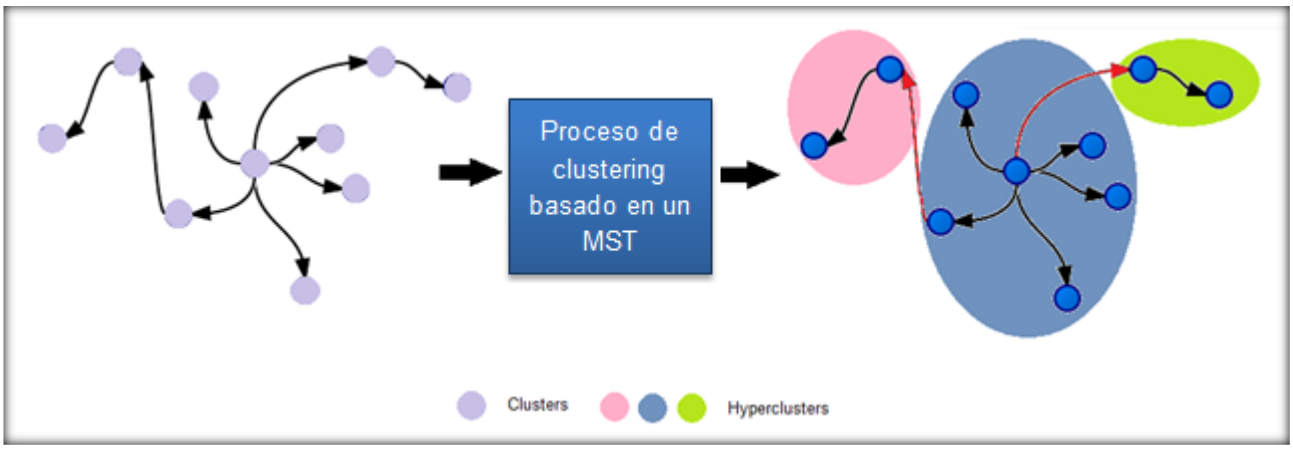


Figura 3-5 : Proceso de clustering basado en un MST que toma como entrada los clusters de genes junto con las relaciones intercluster y como salida proporciona hyperclusters. En la salida se pueden observar los tres hyperclusters obtenidos junto con las aristas interhypercluster proporcionadas por el MST, marcadas en rojo.

Una vez obtenidos los hyperclusters, compruebo si tengo más de 40. En este caso, pude comprobar que si se obtienen más de 40 hyperclusters el programa obtendrá mejores resultados si calcula las particiones de la microarray a partir de un nivel de clusters mayor (hyperclusters de segundo nivel). Resalto de nuevo que esta comparación es ajena al proceso de clustering, y su cometido es optimizar el cálculo de las particiones de la microarray.

Así, si el número de hyperclusters pertenecientes a la microarray es menor a 40, procederé a realizar las particiones de la microarray basándome en estos hyperclusters. En caso contrario y, de forma análoga al proceso de clustering explicado anteriormente, calcularé los hyperclusters de segundo orden.

En caso que haya sido necesario realizar el cálculo de hyperclusters de segundo nivel, la partición de microarrays se efectuará mediante estos cluster de nivel 2.

3.4.3 Proceso de partición de la microarray

Como resultado del proceso de clustering de genes por la correlación entre sus expresiones obtengo los clusters, hyperclusters o hyperclusters de segundo nivel en los que se distribuyen los genes de la microarray. Por otra parte, obtengo también las relaciones intercluster que enlazan éstos hyperclusters de nivel n. Al igual que el caso de las aristas intercluster que enlazan clusters de genes, las relaciones intercluster de nivel n no son otra cosa que aristas del MST que unen genes pertenecientes a diferentes clusters de nivel n.

Las particiones de la microarray serán entonces la selección de los genes de los diferentes clusters de nivel n. Para hallar entonces estas particiones, necesitaré saber qué clusters he de unir. Por otra parte, los applets pueden soportar una media de unos 5.000 genes, de modo que será necesario comprobar que las particiones resultantes no superen en exceso este máximo. . De este modo, la idea principal del algoritmo de partición será ir asignando clusters al applet por grado de correlación con los clusters previamente asignados, hasta llegar al máximo de genes que permita el applet.

Los requisitos que han de cumplir las particiones son los siguientes:

1. Se quieren mostrar unos 5.000 genes de media
2. Los genes han de tener la mayor correlación posible entre todos ellos. Esto nos lo garantiza el anterior algoritmo basado en buscar los clusters de nivel y sus relaciones intercluster. Dado que el objetivo de los clusters es agrupar los genes mejor correlacionados y las particiones de la microarray se basan en los clusters, de esta forma nos estamos asegurando que los genes de cada partición estarán más correlacionados entre ellos que con los genes de las particiones restantes.

Por otra parte, y como he comentado anteriormente, en este proceso es donde toman significado las cotas del número de clusters y de hyperclusters explicadas en la sección 3.4.2. Por una parte es más factible unir 20 clusters de nivel n+1 que 200 de nivel n. No obstante, dado que los genes de las microarrays de gran tamaño están muy correlacionados, tampoco sería posible, en ciertos casos, calcular las particiones directamente a partir de sus hyperclusters de segundo nivel. Si lo hiciera así encontraría un gran número de hyperclusters de segundo nivel cuyo número de genes superaría los 5.000.

Para hallar entonces las particiones de la microarray parto de la siguiente información referente a cada cluster de nivel n obtenido del MST:

- Cluster i de nivel n:
    - Número de genes que lo compone
    - Tabla de relaciones:
        - Relacionado con el cluster j de nivel n mediante una correlación n_1.
        - Relacionada con el cluster k de nivel n mediante una correlación n_2.
        - …

De este modo, el proceso a seguir empieza por buscar la mejor relación que mantienen dos clusters de nivel n. Una vez obtengo dicha relación, compruebo si los dos clusters podrían unirse, es decir, si la suma de sus genes no supera los 5.000 genes. En caso negativo, vuelvo a repetir el proceso hasta hallar dos clusters que pueda unir. Por el contrario, si la unión de los clusters de nivel n tiene menos de 5.000 genes, los pongo en una partición y partiendo de la mejor correlación de estos clusters de nivel n con un cluster ajeno a los incluidos en la partición, selecciono un nuevo cluster de nivel n. Si la suma de los genes de la partición tras añadir este nuevo cluster es menor que 5.000, introduzco el cluster de nivel n seleccionado a la partición. En caso contrario, selecciono otro cluster de nivel n a partir de la siguiente mejor correlación entre clusters de la partición con clusters ajenos a esta. En caso que haya agotado las relaciones existentes de una partición y ésta, no supere los 4.000 genes, la desecho y empiezo el proceso de nuevo. Finalmente, cada vez que una partición o un cluster de nivel n es desechado, se eliminan todas las relaciones con los clusters de nivel n restantes que hayan sido utilizadas. De esta forma me aseguro que no se vuelvan a formar particiones que han sido desechadas previamente.

Una casuística común es que una vez obtenidas las diversas particiones de una microarray, aún queden clusters de nivel n que no pertenezcan a ninguna partición. Dichos clusters reciben el nombre de clusters huérfanos. Un ejemplo de clusters huérfanos puede observarse en la figura 3-6.

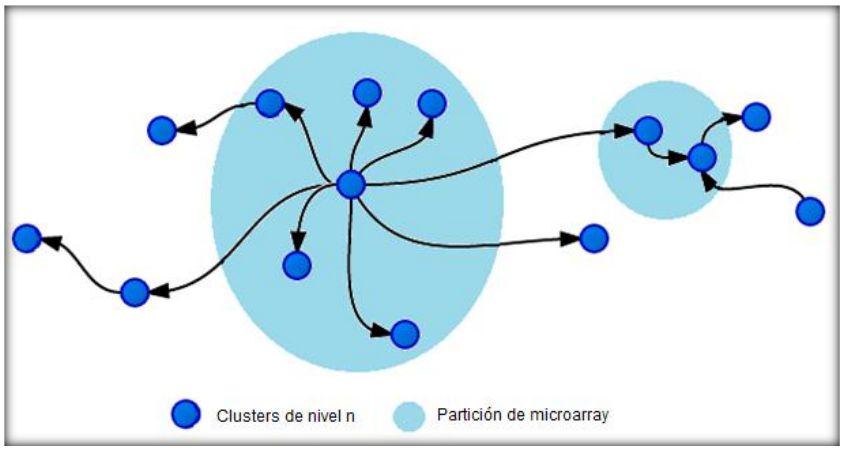


Figura 3-6: Ejemplo de clusters de nivel n huérfanos. Como se puede observar, tras haber encontrado todas las particiones de la microarray posibles, han quedado seis clusters huérfanos. Dichos clusters no podrían ser unidos en una nueva partición siguiendo el procedimiento anterior ya que no existe ninguna arista que los una..

Para solucionar los casos de clusters huérfanos, una vez he distribuido en particiones todos los clusters de nivel n posibles, compruebo si quedan clusters huérfanos. En este caso, vuelvo a lanzar todo el proceso desde la búsqueda del gen mejor correlacionado con cada gen considerando solo estos clusters huérfanos como si se tratase de una nueva microarray. Como se puede observar, se trata de un proceso iterativo.

Finalmente, es posible que ninguna de las uniones de dos clusters tenga un número de genes menor que 4.000. En tal caso, creo una partición de microarray que contenga únicamente el cluster mayor y posteriormente vuelvo a lanzar todo el proceso considerando los clusters restantes como una nueva microarray.

Es posible que ninguna de las aristas intracluster una un par de clusters de la microarray que sumen un número de genes menor a 5.000. En tal caso, creo una partición de microarray que contenga únicamente el cluster con mayor número de genes y posteriormente vuelvo a lanzar todo el proceso considerando los clusters huérfanos como una nueva microarray. En el proceso de clustering de esta nueva microarray, al omitir el cluster con mayor número de genes, es posible que los nuevos clusters que se formen sean diferentes de los anteriores y tengan por tanto un número de genes diferente. De no ser así, se repetirá el proceso volviendo a crear una partición con el cluster de mayor tamaño y se volverá a lanzar todo el proceso considerando los nuevos clusters huérfanos como una nueva microarray. Estas acciones se repetirán hasta obtener todas las particiones de la microarray o bien hasta poder unir dos clusters.

Para facilitar la comprensión de este proceso, a continuación expongo un ejemplo de cómo encontrar las particiones de una microarray.

Tabla: Clusters de nivel n		
cluster 1 de nivel n	Genes: 589	
	Rel.Intercluster:	2 → 0.02541
cluster 2 de nivel n	Genes: 2.840	
	Rel.Intercluster:	1 → 0.02541
		3 → 0.00169
		4 → 0.01276

cluster 3 de nivel n	Genes: 2.922	
	Rel.Intercluster:	2 → 0.00169
cluster 4 de nivel n	Genes: 1.635	
	Rel.Intercluster:	2 → 0.01276
		5 → 0.01897
cluster 5 de nivel n	Genes: 241	
	Rel.Intercluster:	4 → 0.01897

En esta tabla se encuentran cinco clusters de nivel n (clusters, hyperclusters o hyperclusters de segundo nivel, según la correlación que mantengan y el número de genes) que agrupan el total de genes de la microarray a estudiar. En la segunda columna se puede apreciar el número de genes que tiene cada cluster de nivel n así como las relaciones intercluster que mantiene con los clusters restantes.

Pasos:

1. Buscamos la menor relación intercluster, en este caso la que mantienen los clusters 2 y 3 de nivel n. La suma de los genes de estas correlaciones asciende a 5.762 genes, por lo que no es posible crear una partición. La tabla quedaría ahora de la siguiente manera:

Tabla: Clusters de nivel n		
cluster 1 de nivel n	Genes: 589	
	Rel.Intercluster:	2 → 0.02541
cluster 2 de nivel n	Genes: 2.840	
	Rel.Intercluster:	1 → 0.02541
		4 → 0.01276
cluster 3 de nivel n	Genes: 2.922	
	Rel.Intercluster:	-
cluster 4 de nivel n	Genes: 1.635	
	Rel.Intercluster:	2 → 0.01276
		5 → 0.01897
cluster 5 de nivel n	Genes: 241	
	Rel.Intercluster:	4 → 0.01897

2. La siguiente relación es la que mantienen los clusters 2 y 4 de nivel n. En este caso, la suma de genes asciende a 4.475 genes. Dado que esta cantidad es inferior a 5.000, creamos una nueva partición que contenga dichos clusters.
3. El siguiente paso es rellenar la partición. Para ello tenemos dos relaciones posibles, la que mantienen los clusters 1 y 2 de nivel n con correlación 0. 02541, y la que mantienen el 4 con el 5, con correlación 0. 01897. Dado que ésta última relación es mejor, metemos el cluster 5 de nivel n en nuestra partición. Cabe resaltar que podemos realizar esta operación ya que el número de genes asciende a 4.716.

4. En este punto, la tabla de clusters de nivel n tiene el siguiente aspecto:

Tabla: Clusters de nivel n		
cluster 1 de nivel n	Genes: 589	
	Rel.Intercluster:	2 → 0.02541
cluster 2 de nivel n	Genes: 2.840	
	Rel.Intercluster:	1 → 0.02541

		4 → 0.01276
cluster 3 de nivel n	Genes: 2.922	
	Rel.Intercluster:	-
cluster 4 de nivel n	Genes: 1.635	
	Rel.Intercluster:	2 → 0.01276
		5 → 0.01897
cluster 5 de nivel n	Genes: 241	
	Rel.Intercluster:	4 → 0.01897

La única relación intercluster posible es la que mantienen el cluster 1 de nivel n y nuestra partición. No obstante, el número de genes resultante ascendería a 5.305, por lo que no es posible incluir este cluster. De esta forma, dado que nuestra partición tiene más de 4.000 genes, la podemos aceptar. Finalmente, volveríamos a lanzar todo el proceso considerando la unión del cluster 1 y 3 de nivel n como una nueva microarray.

3.4.4 Separación de los ficheros que necesita el applet para las diversas particiones

Como he comentado anteriormente, el prepoceso para el tratamiento de microarrays consta de las siguientes fases:

1. Cálculo de las correlaciones entre genes y determinación de las relaciones de expresión no lineales entre genes.
2. Búsqueda de los genes mejor correlacionados
3. Búsqueda del gen mejor correlacionado con cada gen
4. Cálculo del minimun spannig tree entre los genes de la microarray
5. Proceso de clustering de genes por la correlación entre sus expresiones
6. Cálculo del layout para cada partición concreta

Cada una de estas fases genera unos ficheros diferentes que son necesarios para las operaciones posteriores. Obviamente, los ficheros resultantes de los procesos anteriores a la partición de la microarray, contienen información de la totalidad de la microarray.

Por otra parte uno de los objetivos principales del proyecto es poder tratar cada partición de microarray como una microarray completa. Para ello, es necesario dividir los ficheros resultantes de procesos anteriores y darles la estructura necesaria.

Concretamente tendré los siguientes ficheros:

- Fichero de genes: En este fichero aparece un listado con los nombres de cada uno de los genes que componen la microarray.
- Fichero de correlaciones: En este fichero se encuentran todos los genes junto con las correlaciones que mantienen con los genes restantes. Este fichero sigue el formato de la figura 3-2.
- Fichero de máximas correlaciones: Aquí se encuentran los genes mejor correlacionados con cada gen.
- Fichero del MST: Como su nombre indica, en este fichero se encuentra el minimun spaning tree entre los genes de la microarray.
- Ficheros de relaciones de expresión no lineales: En este caso tendré dos ficheros, uno que contendrá las relaciones de expresión no lineales de baja correlación de todos los genes y otro en el que encontraré las de alta correlación.

- Ficheros de clusters genes basados en las relaciones de expresión no lineales de cada gen: Finalmente en este caso tendré varios ficheros que contienen los clusters de los genes en función con el tipo de relaciones de expresión que mantienen con el resto.

En este punto, simplemente dividiré estos ficheros basándome en las particiones a las que han sido asignados los diferentes genes.

Por otra parte, para poder considerar los nuevos ficheros de particiones como pertenecientes a microarrays completas es necesario un paso intermedio; la conversión de los identificadores de genes. El problema reside en que en una microarray de n genes, encontraríamos todos y cada uno de los genes comprendidos entre el 1 y el n. En una partición, por el contrario, los genes no siguen ningún orden. Es posible que en una partición de 4.000 genes encontremos el gen 5.000, el 6.011, etc, ya que los genes se reparten por las particiones por su grado de correlación con el resto de genes de la partición.

Por este motivo, es necesario reenumerar los genes de todas y cada una de las particiones antes de crear las divisiones de los ficheros. Con esta reenumeración crearé otros ficheros que me servirán de tablas conversoras para llamadas a operaciones externas desde el applet, que necesitarán recuperar el identificador original del gen.

3.4.5 Generación del layout para cada partición concreta

En este momento tengo los ficheros separados de las x particiones en las que se ha dividido la microarray. Dado que anteriormente he reenumerado los genes, puedo aplicar el proceso de layout, explicado en el apartado 3.3.6, a cada partición por separado.

Este proceso entenderá las particiones como si fueran microarrays completas y proporcionará secuencialmente los layouts de cada partición.

Cabe destacar, que dada la gran correlación de los genes en algunas microarrays de gran tamaño, había casos en los que las distancias entre nodos, basadas en correlaciones entre los genes, eran tan insignificantes que no se podían apreciar. Por este motivo, al mostrar el layout en el applet encontraba casos en los cuales era necesario realizar un gran zoom para distinguir los genes.

Para solucionar esto diseñé un algoritmo que, en la última fase del layout, redimensionaba las coordenadas para ocupar el máximo tamaño de pantalla posible. De esta forma, pese que los genes estén muy correlacionados las distancias entre ellos son fácilmente apreciables.

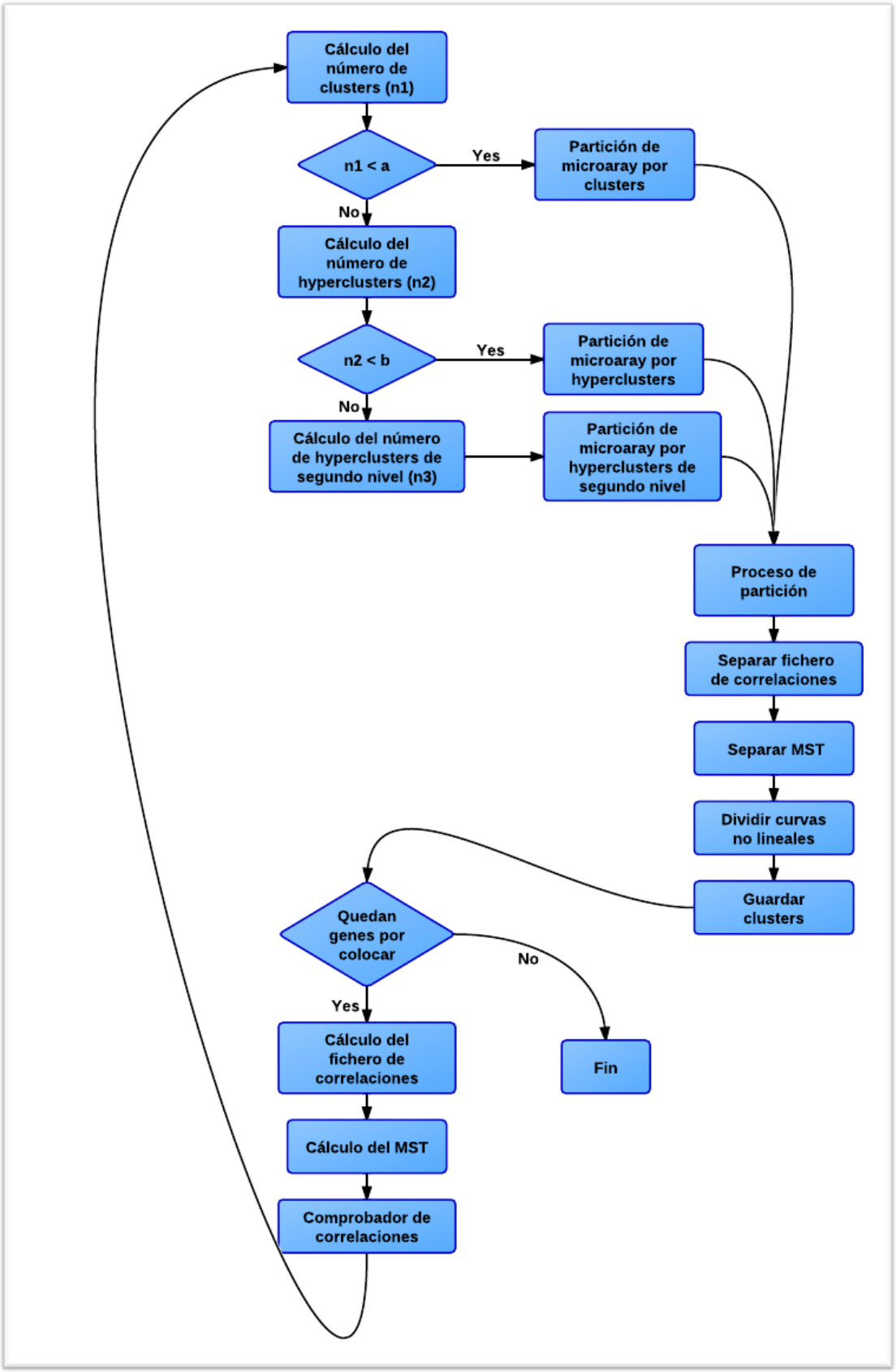


Figura 3-7: Algoritmo del proceso de partición de una microarray. Como se puede observar es un proceso secuencial que dada una microarray proporciona como salida todos los ficheros necesarios para su posterior tratamiento en el applet.

3.5 Adaptación del applet

En la cuarta fase realicé las adaptaciones necesarias al applet para que éste fuera capaz de operar con las diferentes particiones de la microarray. En esta fase fue necesario el estudio de las distintas aplicaciones externas que lanza el applet y la posterior adaptación de algunas de ellas.

Con el fin de que el applet pudiera tratar las particiones de las microarrays realicé dos tipos de modificaciones. Por una parte, las referentes a ficheros y por otra las referentes a genes.

Cuando en el preproceso se crean todos los ficheros de una partición de microarray, estos se reenumeran de forma automática. Es decir, si por ejemplo hablamos de la partición número uno de la microarray m17, los ficheros resultantes serán:

- 17_1.factors
- 17_1.clusters
- 17_1.genes
- BioMST_1.net
- BioFactors2K5_1.net
- relbyGen_1.txt
- Etc

Como se puede observar, los ficheros incluyen en su nombre la partición de la microarray a la que pertenecen. Este hecho facilita su identificación dado que todos los ficheros de una microarray se encuentran en el mismo directorio.

De esta forma, la modificación referente a ficheros consiste básicamente en identificar los casos en los que se trabaja con particiones en lugar de con microarrays completas para modificar el nombre de los ficheros a los que ha de acceder el applet. Así, cada vez que el applet haya de leer un fichero comprueba si está trabajando con una partición o con una microarray completa y según sea, llamará a unos ficheros o a otros.

Por otra parte, como he comentado anteriormente lo genes de las particiones están reenumerados; el identificador de gen que presentan en la partición no se corresponde con el identificador de gen que tienen en la microarray. Así, de no tratar este hecho, estaría incurriendo en un error en muchas de las aplicaciones que ofrece el applet.

El problema es que diversas aplicaciones del applet trabajan directamente con el identificador de los genes para calcular sus respuestas, identificador que les pasa el applet, y si estos identificadores de genes no son correctos, obviamente las respuestas tampoco lo serán. Concretamente, las aplicaciones que necesitan los identificadores de genes son:

- Correlations: Muestra las correlaciones que los genes seleccionados mantienen con el resto de genes de la microarray.
- Dependece analisy: Las fluctuaciones de la dependencia de expresión entre el conjunto de genes suministrados también pueden ser analizadas. En esta aplicación se muestran estas fluctuaciones en una gráfica, en la cual el eje de ordenadas indica el nivel de expresión y el de abscisas indica el parámetro de la relación. Las líneas representan el nivel de expresión de los genes en comparación.

- Get external info: Cuando se ha buscado información externa al applet, ya sea en otras bases de datos o en otras investigaciones, se activa esta opción que permite marcar en el grafo los genes resultantes de la consulta.

Para solucionar este problema, cuando se llama a una de las aplicaciones anteriores compruebo si el applet está trabajando con una partición o con una microarray completa. En caso que esté trabajando con una partición, antes de calcular los resultados pertinentes, realizo una conversión de identificadores mediante un fichero generado en el proceso de partición de la microarray. Este fichero es diferente para cada partición y contiene para cada gen, su identificador de gen en la partición y su identificador de gen en la microarray. Una vez obtengo entonces los identificadores de genes correctos, procedo a realizar los cálculos que sean necesarios.

De esta forma me aseguro que los identificadores de genes usados son los correctos y por tanto los resultados mostrados también lo serán.

En el caso de la opción Get external info hay una pequeña modificación y es que dicha opción puede devolver información relativa a toda la microarray y no solo a la partición con la que se trabaja. Por este motivo, una vez realizada la conversión de identificadores de genes es necesario seleccionar únicamente los genes pertenecientes a la partición seleccionada para poder mostrarlos así en el grafo.

Cabe destacar que el resto de procesos que realiza el applet son independientes de los identificadores de genes de la microarray ya que o bien utilizan el nombre de los genes, como es el caso de las aplicaciones Mspath o Correlations table, o bien utilizan directamente identificadores de genes de partición. Éste el caso de los procesos que tratan con el Layout, el MST o las relaciones de expresión no lineales.

3.6 Filtrado de relaciones de expresión no lineales

Para la realización de esta fase dispongo de una serie de programas que, a partir de los datos obtenidos por microarray, calculan las Principal curves of Oriented Points (PCOP) [9]. Posteriormente, a partir de estas PCOP se analizan las curvas obteniendo los máximos, los mínimos y los puntos de inflexión de cada una y se crean diferentes tipos clusters según diversas condiciones referentes a estas relaciones de expresión no lineales. Obtener esto es importante si se quieren comparar unas curvas con otras. De esta forma es posible determinar qué tipos de relaciones diferentes se dan entre los genes a nivel de expresión y, a partir de éstas, clasificar los genes por el tipo de relaciones que mantiene con el resto.

Dado el gran volumen de genes con el que he trabajado, he podido apreciar que el número de relaciones de expresión no lineales entre los genes se dispara. Por este motivo ha sido necesaria una quinta fase. En ésta he diseñado e implementado dos algoritmos para el filtrado de relaciones de expresión no lineales entre los diferentes genes; uno mientras estas relaciones se calculan y otro para restringir las relaciones de expresión no lineales que se mostrarán en el applet. De esta forma he obtenido dos grandes ventajas; por un lado una visualización más enfocada y clara de la información y por otro, he descartado posibles problemas posteriores debidos a la sobrecarga de datos.

Tal y como he comentado anteriormente, el problema viene dado por el hecho de que en microarrays de gran tamaño, los genes suelen provenir del mismo proceso biológico. Por este motivo, es normal que los

genes estén más correlacionados. Por otra parte, al aumentar el número de genes obviamente aumenta el número de relaciones entre estos.

3.6.1 Filtrado de relaciones de expresión no lineales durante la detección de las mismas

Los hechos expuestos anteriormente hacen que los filtros existentes para detectar si la relación de expresión entre dos genes es lineal o no para microarrays de tamaño reducido, queden obsoletos para estas nuevas microarrays de gran tamaño.

Concretamente, la fórmula que utilizaban para filtrar estas relaciones era la siguiente:

$$0.08 \cdot \left(\frac{1.500}{número\ de\ genes}\right)^{0.6}$$

Si el valor de la relación de expresión entre dos genes obtenido era menor que el valor del filtro, la relación se consideraba como no lineal.

A continuación se puede encontrar una tabla con los valores del filtro para diversos tamaños de microarrays:

Número de genes	Valor del filtro
673	0,12940043
1.600	0,07696136
14.000	0,02094440
16.000	0,01933182
20.000	0,01690934
27.500	0,01396835
30.000	0,01325782

Como se puede observar, para microarrays de gran tamaño el valor del filtro es excesivamente alto y dejaría pasar un gran número de relaciones de expresión no lineales. Para solucionar estos casos modifiqué la fórmula utilizada por la siguiente:

$$0.12 \cdot \left(\frac{1.600}{número\ de\ genes}\right) - \left(\frac{número\ de\ genes}{40.000}\right)^{18}$$

El significado es el mismo; si el valor de la relación de expresión entre dos genes obtenido es menor que el valor del filtro, la relación se considera como no lineal.

Utilizando esta fórmula, la tabla anterior quedaría de la siguiente manera:

Número de genes	Valor del filtro
673	0,28528975
1.600	0,12000000

14.000	0,01371428
16.000	0,01199993
20.000	0,00959619
27.500	0,00580446
30.000	0,00076229

Como se puede observar, el nuevo filtro es mucho más restrictivo que el anterior para microarrays de gran tamaño. No obstante, pese aplicar este nuevo filtro seguía obteniendo demasiadas relaciones de expresión no lineales. Por este motivo decidí modificar otro valor; el ángulo de curvatura. Este ángulo fija el valor a partir del cual se puede considerar una curva como recta. Cuanto mayor sea este ángulo menor número de curvas serán obtenidas.

En la aplicación anterior, el ángulo de curvatura estaba fijado a 159 grados, un valor excesivamente alto para microarrays de gran tamaño. Para restringir aún más la creación de relaciones de expresión no lineales decidí modificar éste parámetro y hacerlo dependiente del número de genes de la microarray. A continuación se muestra la fórmula utilizada.

$$160 - \left\lfloor \frac{\frac{15}{20.000} + \frac{14}{18.400}}{2} \cdot número\ de\ genes \right\rfloor$$

donde |x| representa la parte entera del número x. En la siguiente tabla se muestran los valores de los ángulos para diversos números de genes:

Número de genes	ángulo
673	160
1.600	159
14.000	150
16.000	148
20.000	145
27.500	140
30.000	138

Como se puede observar, para microarrays de tamaño reducido el valor del ángulo prácticamente es el mismo que en el caso original mientras que para grandes valores de genes, el ángulo se reduce de forma notable.

No obstante, pese a la modificación de las fórmulas para el filtrado de relaciones de expresión no lineales, pude comprobar la existencia de casos en los cuales seguía apareciendo un número excesivo de curvas.

Por otra parte, cada curva puede generar de dos a seis ficheros diferentes. De esta forma, en caso de obtener muchas curvas, es posible sobrecargar el server de información y hacer que falle. Por este motivo, decidí diseñar e implementar un algoritmo que permite restringir el número de ficheros creados, restringiendo por tanto el número de curvas obtenidas.

Este algoritmo se ejecuta mientras se calculan las relaciones lineales y consigue restringir el número de ficheros creados a tan solo 1.000.000, cantidad que el server puede soportar sin problema.

Para comprender su funcionamiento, es necesario resaltar que las curvas se calculan a partir del fichero de correlaciones de todos los genes, la estructura del cual se puede observar en la figura 3-2. Tras hacer esta reflexión, el algoritmo implementado es el siguiente:

Filtrado_Nolineal

```
Max = 1.000.000
Margen = 50
Numero_de_lineas_totales = CalcularLineas(numero_de_genes)
Valor_teorico_por_linea = MAX / Numero_de_lineas_totales

ComprobarFicheros(n, curvas){
        Valor_teorico = Valor_teorico_por_linea * CalcularLineas(n)
        Si (Curvas < Valor_teorico + margen){
                Restringir_filtros();
                BorrarFicherosErroneos();
        }
}

Algoritmo_Principal{
        Curvas=0
        Para cada i desde 1 hasta numero_de_genes{
                Para cada j desde i+1 hasta numero_de_genes{
                        CalculoDeCurvas();
                        Si se han creado n curvas:
                                Curvas = Curvas + n
                }
                ComprobarFicheros(i, Curvas)
        }
}
```

Dónde:

- CalculoDeCurvas() es el proceso que decide si la relación que une dos genes es una curva o no.
- CalcularLineas(n) es un proceso que devuelve el número de líneas del fichero de correlaciones de todos los genes anteriores a la aparición del gen n.
- RestringirFiltros() es un proceso que permite restringir el valor actual de los filtros que deciden si una relación es una curva o no.
- BorrarFicherosErroneos() en un proceso que borra todos los fichero creados con correlación superior al actual valor de los filtros.

Como se puede observar, gracias a este último algoritmo me puedo asegurar que el número de ficheros creados nunca va a provocar un fallo en el server. Por otra parte, esta modificación sirve como filtro extra para la creación de relaciones de expresión no lineales.

3.6.2 Filtrado de las relaciones de expresión no lineales mostradas en el applet

A pesar de todo el proceso de filtrado de relaciones de expresión no lineales durante la detección de las mismas, pude comprobar que en el applet se seguían visualizando un número excesivo de curvas. Por este motivo decidí implementar, como último paso del preproceso, un último filtrado de relaciones de expresión no lineales.

Este último filtrado consiste en seleccionar las mejores curvas para mostrarlas en el applet. De esta forma, cuando en el applet se quieran visualizar las curvas más correlacionadas se mostrarán las 2.000 mejores y cuando se quieran visualizar las curvas con un factor de correlación más permisivo, se mostrarán las 4.000 mejores.

Gracias a este filtrado de relaciones de expresión no lineales mostradas en el applet he conseguido que la visualización de dichas relaciones mejore sustancialmente. Ahora, al visualizarse sólo las mejores curvas, la información aparece mucho más ordenada. Por otra parte, al trabajar con menos datos, el applet funciona de una manera más rápida y se consiguen evitar posibles problemas derivados del exceso de carga de datos.

3.7 Adaptación del aplicativo web

Finalmente, la última fase del proyecto consistió en la adaptación del aplicativo web para conseguir mostrar simultáneamente todas las particiones que conforman una microarray.

En la aplicación anterior, dado que las microarrays no se partían, solo se necesitaba mostrar un applet para cada microarray. Ahora cada microarray puede dividirse en x particiones. Así, cuando se quiera visualizar una microarray de gran tamaño será necesario que se abran tantos applets como particiones.

Para la realización de esta fase la primera idea que tuve fue que al seleccionar una microarray se abriera automáticamente un applet por cada partición y se cargaran todos de forma simultánea. El gran problema que me encontré fue que los applets, para cargarse utilizaban la caché de java con lo cual al cargarse todos a la vez, los applets leían información de diversas particiones y fallaban.

Para solucionar este problema decidí crear siete applets, número máximo de particiones que se mostrarán. De esta forma, al seleccionar una microarray de gran tamaño, se abrirá un applet con la primera partición. En cuanto esta se haya cargado por completo, se abrirá una nueva ventana con la siguiente partición y se cargará. Este proceso se repetirá hasta alcanzar el número de particiones en los que se divida la microarray o bien hasta mostrar siete particiones.

Una vez cargadas todas las particiones, el usuario podrá trabajar con ellas de forma simultánea, visualizando y comparando los genes que hayan sido objeto de su estudio.

4. Resultados

4.1 Optimización del cálculo de correlaciones entre genes

Como resultado de las optimizaciones realizadas en el cálculo de las correlaciones entre genes, ahora es posible estudiar microarrays en las que se han omitido las respuestas de diversos genes frente a cualquier condición muestral.

El estudio de este tipo de microarrays omitirá entonces la condición muestral faltante para ese gen en concreto pero seguirá teniendo en cuenta tanto al gen como a las condiciones muestrales restantes.

Este hecho le da un valor añadido a la aplicación ya que en muchos casos y por diversas circunstancias, no es posible obtener la respuesta de todos los genes frente a todas las condiciones muestrales.

4.2 Mejoras en el cálculo de las mejores correlaciones

Gracias a la modificación del proceso de cálculo de las mejores correlaciones he conseguido disminuir significativamente los tiempos de espera. En la siguiente tabla se puede observar la diferencia de tiempo obtenida utilizando los dos procesos para microarrays de distinto tamaño.

Número de genes	Proceso anterior	Nuevo proceso
2.998	5.971 s	2 s
5.000	28.856 s	6 s
7.702	101.248 s = 1,17 días	19 s
14.012	889.056 s = 10,29 días	47 s

Una idea gráfica de este hecho se puede observar visualizando la siguiente imagen:

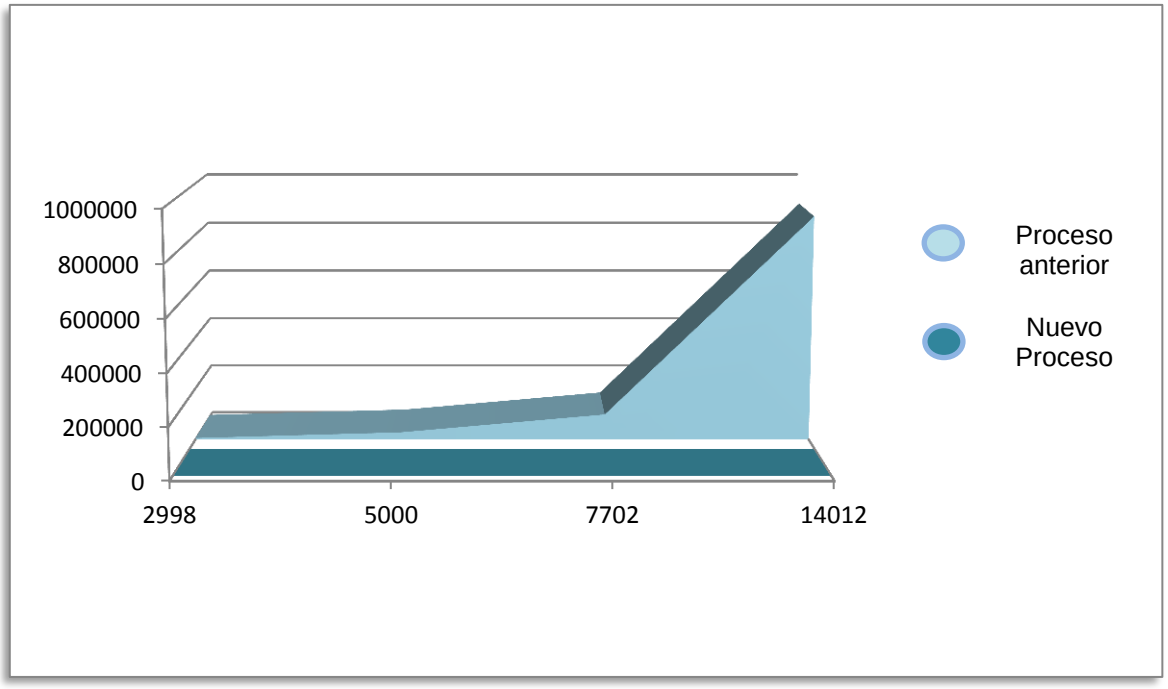


Figura 4-1: Grafico que ilustra las grandes diferencias existentes entre el antiguo y el nuevo proceso de cálculo de mejores correlaciones.

Como se puede observar, el proceso anterior consumía un tiempo que es impermisible, llegando a necesitar varios días para unos cálculos que el nuevo proceso puede efectuar en segundos.

Por este motivo, es obvio que la mejora del proceso de cálculo de las mejores correlaciones ha resultado un éxito.

4.3 Resultados del filtrado de relaciones de expresión no lineales

Como he comentado anteriormente, gracias a los diversos filtros de relaciones de expresión no lineales implementados, he conseguido dos grandes ventajas. Por una parte evito problemas derivados del exceso de carga de ficheros tanto en el server del IBB como en el propio applet y por otra consigo una visualización mucho más nítida y selectiva de este tipo de relaciones.

4.3.1 Resultados del filtrado de relaciones de expresión no lineales durante la detección de las mismas

Como he comentado anteriormente, el número de relaciones de expresión no lineales de una microarray está ligado completamente al grado de correlación que mantienen sus genes. No obstante, pese a que

el tamaño de la microarray influye en la correlación de sus genes, no es un factor determinante para conocer el número de relaciones de expresión no lineales.

Así es posible entender que ninguna fórmula dependiente del número de genes conseguirá que el número de relaciones de expresión no lineales existentes en una microarray no ascienda de un número determinado. Para comprender mejor este hecho, a continuación se muestra una tabla que muestra el número de ficheros generados para diversas microarrays de igual y diferente tamaño.

Número de genes	Filtro utilizado	Curvatura máxima	Ficheros generados
673	0.1	159	45.775
5.000	0.01	159	18.795.572
		120	17.650.942
	0.001	130	11.354.106
			184.315
14.012	0.001	130	1.419.321

En la tabla anterior se muestra por cada microarray tanto el filtro utilizado como la curvatura máxima permitida. El valor del filtro indica que solo se considerarán relaciones de expresión no lineales las que tengan un valor de correlación menor que el que indica el filtro. Por otra parte el parámetro de curvatura máxima fija el valor a partir del cual se puede considerar una curva como recta. Cuanto mayor sea este ángulo menor número de relaciones de expresión no lineales serán obtenidas.

Dado que el número de ficheros generados es directamente proporcional al número de relaciones de expresión no lineales halladas, es posible hacerse una idea de que no existe ninguna fórmula que por sí sola asegure que no se generarán más relaciones de expresión no lineales que un valor prefijado. De hecho, observando la tabla anterior es posible comprobar cómo, en ciertos casos y pese a que dos microarrays tengan el mismo número de genes, pueden existir diferencias importantes en el número de relaciones de expresión no lineales halladas. De la misma manera, es posible que se encuentren más relaciones de expresión no lineales en una microarray más pequeña que el número de relaciones de expresión no lineales halladas en otra de mayor tamaño.

Por este motivo el algoritmo implementado es óptimo para solucionar este problema ya que, al restringir los filtros de forma dinámica, asegura que el número de relaciones de expresión no lineales será aproximadamente de 1.000.000.

4.3.2 Resultados del filtrado de las relaciones de expresión no lineales mostradas en el applet

Por otra parte, ya he comentado anteriormente que pese a utilizar el filtrado de relaciones de expresión no lineales durante la detección de las mismas, el número de relaciones este tipo mostradas en el applet seguía siendo excesivo. Por este motivo diseñé el último filtro de relaciones de expresión no lineales.

En la siguiente figura se puede observar la comparativa visual de los resultados obtenidos con la aplicación del filtro y sin ella.

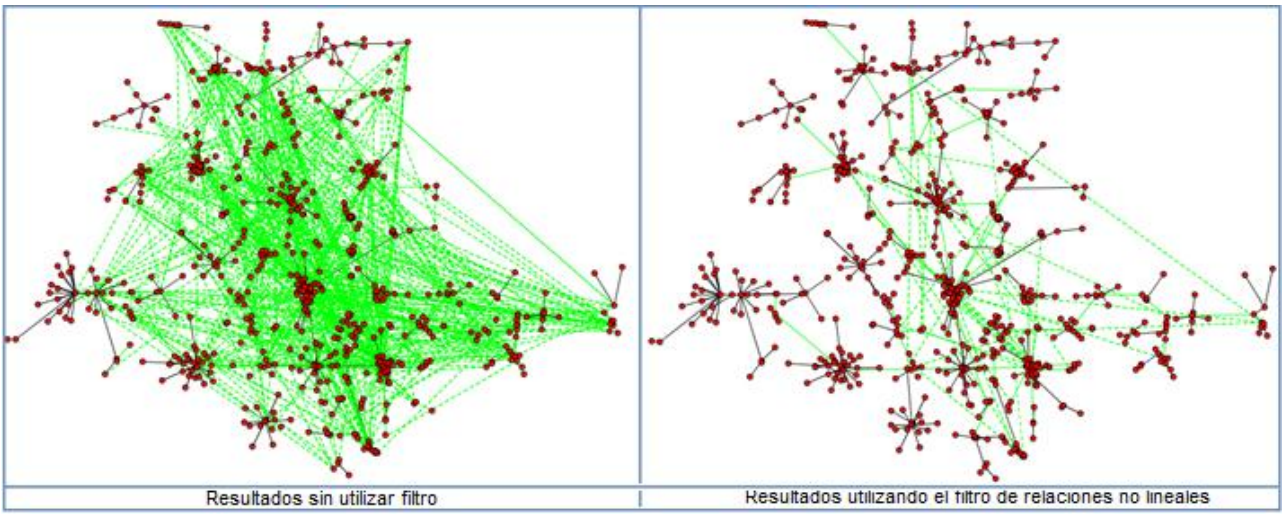


Figura 4-2: Comparativa de resultados usado el filtrado de las relaciones de expresión no lineales mostradas en el applet y omitiéndolo. En la imagen de la izquierda se muestran los resultados obtenidos sin el filtrado de relaciones de expresión no lineales. Por el contrario, en la imagen de la derecha se muestran los resultados obtenidos con la utilización de dicho filtrado.

Como se puede observar, la información mostrada sin este filtro es excesiva. Al mostrarse tantas curvas resulta una tarea imposible de cara al investigador, poder extraer información útil de ellas. De hecho, la simple visualización provoca una sensación de caos y desorden que invita a la ocultación de la misma. Y no es este el único problema, al tener que manejar el applet toda esta información, se puede observar como su funcionalidad se ralentiza. Realizar cualquier operación resulta más pesado si se tiene la visualización activada.

Con la aplicación de este último filtro, por el contrario, al mostrarse únicamente la información más selectiva, el investigador puede trabajar de una forma más cómoda. Por otra parte, consigo también mantener la rapidez del applet al no sobrecargarlo con excesiva información.

4.4 Resultados de la partición de microarrays

Como he comentado anteriormente, el proceso de partición de microarrays está diseñado de forma que dependiendo de la correlación de los genes, y no del tamaño de la microarray, se pueden crear las particiones con los cluster de nivel n óptimos. Es decir, es posible que dadas dos matrices del mismo tamaño, las particiones de una estén basadas en hyperclusters mientras que las de la otra se basen en hyperclusters de segundo nivel.

Este hecho es muy importante por dos motivos. Por una parte, permite asegurar que los genes que se muestran en cada partición mantienen la mejor correlación posible entre ellos. Por otra parte, ahorra mucho tiempo de espera. Esto es debido a que se asegura que las particiones se realicen con el número óptimo de clusters de nivel n; no es lo mismo unir 10 clusters que unir 50.

Finalmente, en caso que en un futuro se quiera trabajar con microarrays de mucho mayor tamaño, el diseño escalable del proceso permite que los cambios a realizar sean mínimos. Éste un punto a favor ya que evita tener que perder tiempo entendiendo algoritmos y rediseñándolos.

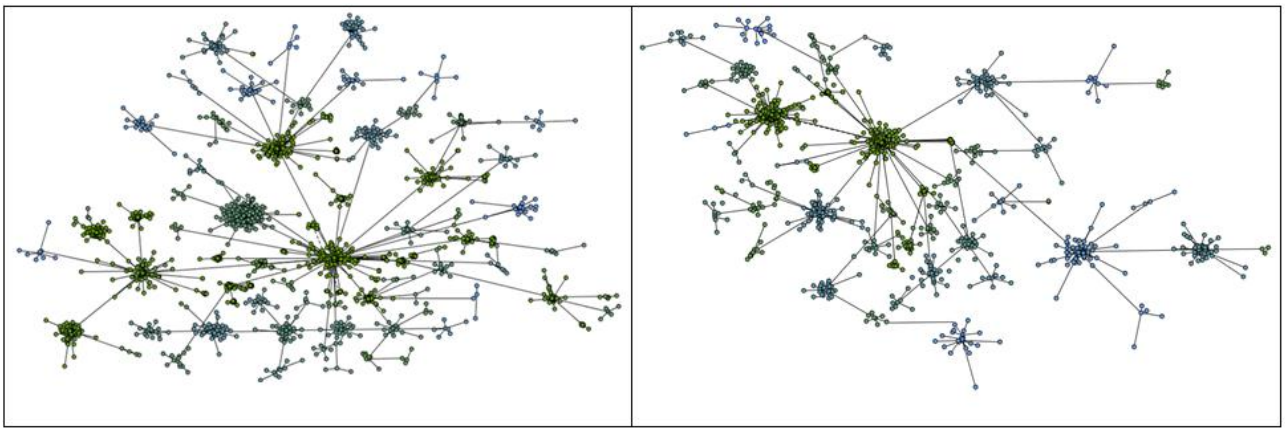


Figura 4-3: Ejemplo de una microarray de gran tamaño expresada como dos particiones de microarrays. En ambas particiones se pueden observar los distintos genes que forman la microarray coloreados según el cluster al que pertenezcan. Se pueden observar también los MST de cada partición.

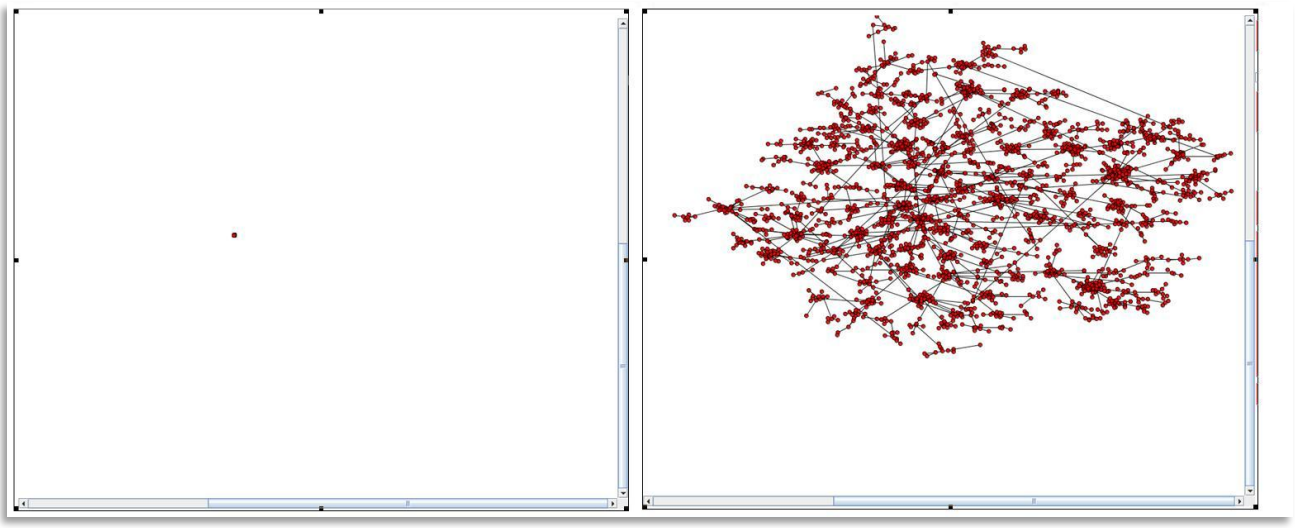
4.5 Resultados de la optimización del cálculo del layout

Como he comentado anteriormente, el layout original estaba pensado para trabajar con genes cuyas correlaciones no fueran excesivamente pequeñas. No obstante, en las microarrays de gran tamaño predominan las bajas correlaciones.

El problema principal es que dicho algoritmo tomaba las correlaciones como distancias y, al existir valores tan pequeños, los resultados eran inapreciables a simple vista.

Gracias a la modificación realizada, el resultado actual es que independientemente del tipo de correlaciones que se tengan, los genes se van a disponer en el applet ocupando todo el espacio que puedan.

A continuación se pueden observar las diferencias obtenidas gracias a la modificación implementada.



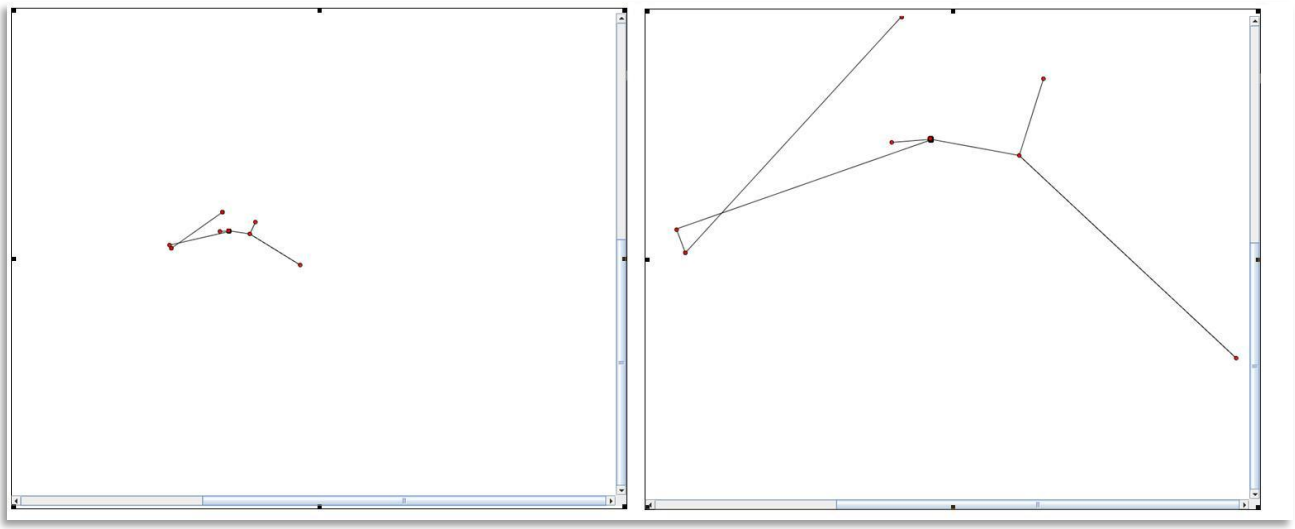


Figura 4-4: Representación de las diferencias tras la modificación del layout. Las imágenes de la izquierda representan los resultados obtenidos originalmente. Las de la derecha, por el contrario, muestran los resultados obtenidos tras realizar la modificación en el layout.

Como se puede apreciar a simple vista, gracias a la modificación realizada los genes aparecen más dispersos en el layout, hecho que facilita sustancialmente su estudio.

4.6 Resultados de la adaptación del aplicativo web

La adaptación realizada al aplicativo web permite, en caso de querer visualizar una microarray de gran tamaño, poder ver a la vez todas las particiones que la conforman.

Tal y como se muestra en la siguiente imagen, una vez seleccionada la microarray, se abrirá una nueva ventana o pestaña por cada partición de la microarray.

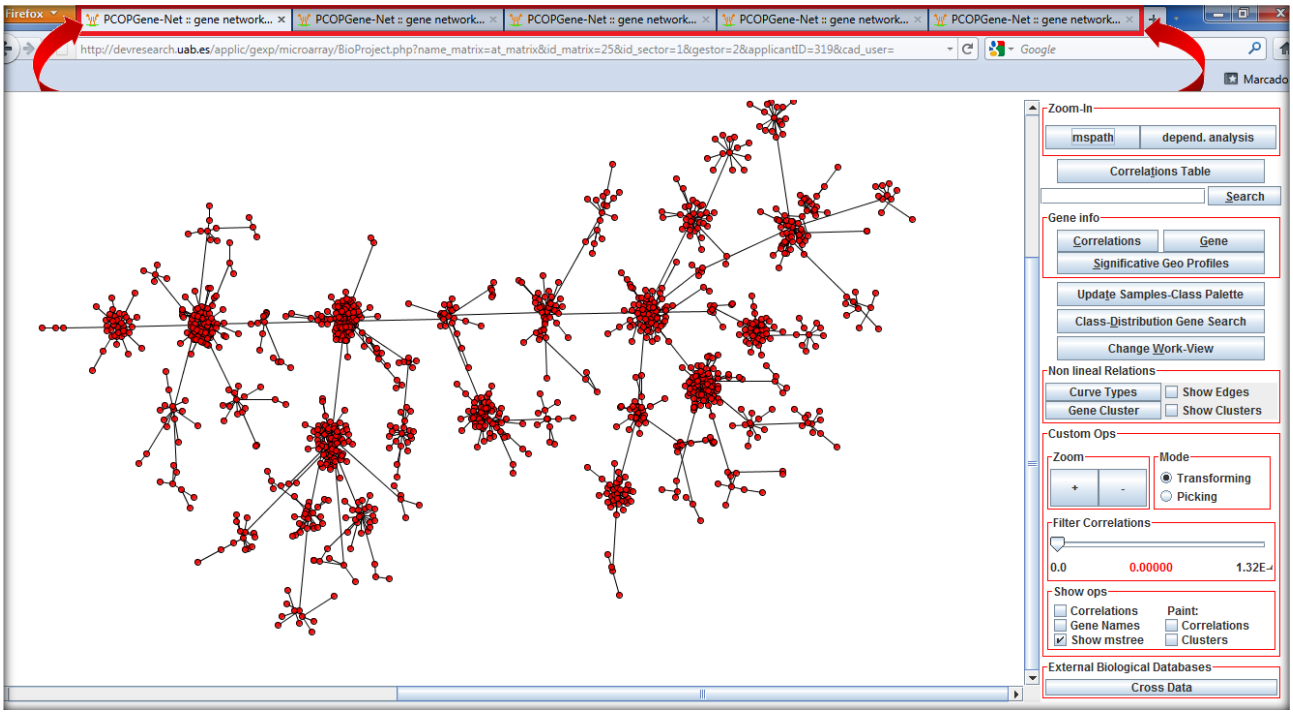


Figura 4-5: Muestra de la visión de una microarray de gran tamaño partida en varios applets. Al usar la aplicación PCOPGene-Net para una microarray de este tipo, se abrirán tantas pestañas o ventanas como particiones conformen la microarray, hasta un máximo de 7.

Una vez abiertas todas las particiones, el usuario podrá utilizar todas las aplicaciones que el applet le proporciona de forma simultánea en todas y cada una de las particiones, cumpliendo así con los objetivos que se proponían en este proyecto.

5. Informe técnico

A continuación describo de manera más técnica por secciones los programas implementados y la estructura de directorios seguida para el desarrollo del proyecto. En los programas implementados realizo una descripción del funcionamiento, su ubicación, cuales son los parámetros de entrada y de salida y los módulos o librerías necesarias para su utilización.

5.1 Estructura de directorios

Entender la estructura de directorios es básico para poder situar correctamente los programas, la información utilizada y las llamadas que se realizan.

A continuación detallo la estructura de directorios donde se encuentran los diversos procesos de este proyecto.

El directorio principal de la parte de preproceso es /cgi-bin/pcop/. Dentro de éste se encuentran todos los programas ejecutables. De forma detallada tenemos:

- fullcorrelations/compile/: En esta carpeta se encuentra el código fuente de todos los procesos. Concretamente:
  - Factors2samples/: código y ejecutables del proceso encargado de tratar los datos para su posterior incorporación en el applet.
  - Lanzadora/: código y ejecutables del proceso general, encargado de llamar a todos los procesos necesarios para el preproceso de la microarray.
  - Correl/: código y ejecutables del proceso que busca las mejores correlaciones de cada gen.
  - Clustering/: código y ejecutables del proceso encargado de buscar los clusters o las particiones de la microarray dependiendo del tamaño de ésta.
  - Prim/: código y ejecutables del proceso encargado de buscar el minimun spaning tree.
  - Grafoclusters/: código y ejecutables del proceso encargado de buscar el layout de la microarray.
  - Comprobador/: código y ejecutables del proceso encargado de comprobar el fichero de mínimas correlaciones de los genes de una microarray.
  - FiltradoNoLineal/: código y ejecutables del último proceso encargado del filtrado de aristas no lineales.
- fullcorrelations/: Este es el directorio desde donde se realizan las llamadas a los diferentes procesos. Por este motivo, en esta carpeta se encuentran todos los archivos ejecutables comentados en el punto anterior.
- microarray/: Carpeta donde se encuentran todas las microarrays que se quieren estudiar, cada una en su correspondiente carpeta. El prepoceso guardará en la carpeta de la microarray que estudie todos los ficheros resultantes.

En el directorio www/applic/gexp/microarray se encuentran tanto el applet como todos los ficheros necesarios para asegurar su correcto funcionamiento.

Para que una microarray x funcione correctamente es necesario que en su directorio (microarray/mx) figuren los siguientes ficheros:

- x. genes: Fichero con los nombres de cada gen de la microarray.
- x.samples: Fichero que contiene la totalidad de la microarray.
- x.snames: Fichero con los nombres de las condiciones muestrales.
- x.factors: Es el fichero que contiene las correlaciones de todos los genes con todos los genes. Su estructura sigue la de la figura 3-2.
- x.maxcorr: Fichero que contiene los genes mejor correlacionados con cada gen.
- BioMST.net: Es el fichero que contiene el MST.
- x.clusters: Fichero que contiene todos los clusters de la microarray junto con las relaciones intercluster.
- BioFactors2K5.net: Fichero que contiene el layout de la microarray.
- /factors: En el directorio factors de la microarray es necesario que haya un fichero por cada gen i, llamado i.txt. Este fichero ha de contener las 500 mejores correlaciones del gen i.
- /nonlineal:
    - /class/relbyGenF.txt: Fichero que contendrá las relaciones de expresión no lineales de baja correlación.
    - /classHeighF: En este directorio han de aparecer los siguientes ficheros y directorios:
        - relbyGenF.txt: Fichero que contendrá las relaciones de expresión no lineales de alta correlación.
        - /clustersSIN/: Directorio en el que se encuentra la información relativa al proceso de clustering de genes basado en las relaciones de expresión no lineales.
            - clusters_normClass.clust: Fichero que contiene el clustering normalizando clases (columnas).
            - /clustersSIN/clusters_normClass_normGen.clust: Fichero que contiene el clustering normalizando clases y gene (filas y columnas).
            - /clustersSIN/clusters_normGen.clust: Fichero que contiene el clustering normalizando genes (filas).
            - /clustersSIN/clusters_sinNormalizar.clust: Fichero que contiene el clustering sin normalizar.

Para el caso de microarrays de gran tamaño se añaden además los siguientes ficheros:

- Conversor.txt: Fichero que contiene la tabla necesaria para convertir los identificadores de genes de las particiones en identificadores de genes de la microarray.
- NumApplets.txt: Este fichero informa del número de particiones que forman la microarray.

Por otra parte, dado que en el mismo directorio se tiene todos los ficheros de las diversas particiones de la microarray, todos los ficheros mencionados en el punto anterior han de incluir en su nombre el número de la partición a la que pertenecen. Así, para el caso de microarrays grandes se tendrían los siguientes ficheros:

- x__i. genes: Fichero con los nombres de cada gen de la partición.
- x_i.factors: Es el fichero que contiene las correlaciones de todos los genes con todos los genes.
- x_i.maxcorr: Fichero que contiene los genes mejor correlacionados con cada gen.
- BioMST_i.net: Es el fichero que contiene el MST.
- x_i.clusters: Fichero que contiene todos los clusters de la partición junto con las relaciones intercluster.

- BioFactors2K5_i.net: Fichero que contiene el layout de la partición.
- /factors: En el directorio factors de la microarray es necesario que haya un fichero por cada gen i, llamado i.txt. Este fichero ha de contener las 500 mejores correlaciones del gen i.
- /nonlineal:
    - /class/relbyGenF_i.txt: Fichero que contendrá las relaciones de expresión no lineales de baja correlación.
    - /classHeighF: En este directorio han de aparecer los siguientes ficheros y directorios:
        - relbyGenF.txt: Fichero que contendrá las relaciones de expresión no lineales de alta correlación.
        - /clustersSIN/: Directorio en el que se encuentra la información relativa al proceso de clustering de genes basado en las relaciones de expresión no lineales.
            - clusters_normClass_i.clust: Fichero que contiene el clustering normalizando clases (columnas).
            - /clustersSIN/clusters_normClass_normGen_i.clust: Fichero que contiene el clustering normalizando clases y gene (filas y columnas).
            - /clustersSIN/clusters_normGen_i.clust: Fichero que contiene el clustering normalizando genes (filas).
            - /clustersSIN/clusters_sinNormalizar_i.clust: Fichero que contiene el clustering sin normalizar.

Donde i irá desde 1 hasta en número de particiones que contenga la microarray.

No obstante, cabe remarcar que no es necesario preocuparse por todo el repertorio de ficheros. Dados únicamente los ficheros x.genes, x.samples y x.snames, el preproceso realiza todos los cálculos y es el encargado de generar todos los ficheros necesarios para el buen funcionamiento de la aplicación.

5.2 Estructura de los ficheros

A continuación detallo la estructura que han de seguir los ficheros que servirán de entrada o salida de los programas que componen el preproceso.

5.2.1 Fichero x.genes

El fichero x.genes contiene los nombres de cada gen de la microrray, ordenados por su identificador, es decir, por orden de aparición en las columnas de la microarray.

5.2.2 Fichero x.samples

El fichero x.samples contiene la totalidad de la microarray. Las filas del fichero representan los genes y las columnas las condiciones muestrales. De esta forma en la posición i, j se encuentra la respuesta del gen i a la condición muestral j.

Cabe resaltar que en todas las filas los elementos están separados por tabulaciones.

5.2.3 Fichero x.snames

En el fichero x.snames se encuentran los nombres de las condiciones muestrales utilizadas en la microarray x. Estos nombres se encuentran ordenados según su aparición en la microarray, es decir, en la fila i del fichero x.snames se encuentra el nombre de la condición muestral que figura en la columna i de la microarray.

5.2.4 Fichero x.factors

Como he comentado anteriormente, el fichero x.factors contiene las correlaciones de todos los genes con todos los genes. Su estructura sigue la de la figura 3-2 y sus elementos están separados por espacios.

Gracias a este fichero es posible conocer la correlación que mantiene cualquier gen i con cualquier otro gen j.

5.2.5 Fichero x.maxcor

El fichero x.maxcorr contiene un listado con la mejor correlación de cada gen. Su estructura se puede observar en la siguiente figura:

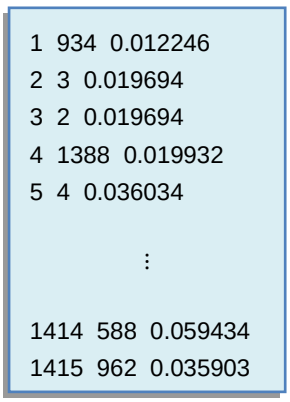


Figura 5-1: Ejemplo de fichero x.maxcor. En la primera posición se encuentra el gen a estudiar, a continuación el gen con el que mejor correlación mantiene y por último aparece dicha correlación, todos separados por un espacio.

5.2.6 Fichero BioMST.net

En el fichero BioMST.net se encuentra el resultado del mínimum spannig tree de la microarray. De hecho, aparece una arista del MST en cada línea del fichero que contiene los dos genes que une y la correlación que los mismos mantienen. Dichos campos están separados por un espacio cada uno.

5.2.7 Fichero x.clusters

En el fichero x.clusters se encuentran todos los clusters que forman la microarray separados por dos ceros. Cada cluster se expresa indicando las aristas del MST que unen dos genes pertenecientes a dicho cluster.

A continuación, seguidas de un cero y un uno figuran todas las aristas intercluster, es decir, las aristas del MST que pertenecen a genes de clusters diferentes.

La estructura de este fichero se puede observar en la siguiente figura:

```
963 964 0.008489
964 963 0.008489
0 0
1262 1270 0.057151
1270 1269 0.029361
1265 1270 0.069290
1269 1270 0.029361
0 0
        ⋮
0 1
3 40 0.041578
5 456 0.042827
21 297 0.047887
        ⋮
```

Figura 5-2: Ejemplo de fichero x.clusters. En este fichero se encuentan incicialmente los diversos clusters de la microarray separados con dos ceros y expresados como aristas del MST que contiene dos genes del cluster. A continuación, separadas por un cero y un uno se encuentran las aristas intercluster.

5.2.8 Fichero BioFactors2K5.net

En el fichero BioFactors2K5.net se encuentra el layout de la microarray. En la primera línea del fichero se encuentra el número de genes que componen la microarray. En las siguientes líneas se encuentran los identificadores de los genes seguidos de sus nombres entre comillas y de las coordenadas x e y que los situarán en el applet (una línea por gen). A continuación, en la misma línea figura el color que tendrá el gen expresado como un número.

Separadas mediante una línea que contiene la expresión **Edges* figuran las diferentes aristas que componen el grafo.

En la siguiente figura se puede observar un ejemplo de fichero BioFactors2K5.net.

```
*Vertices 1416
1 "Not found EST code." 457.116211 245.042740 0.059384
2 "SCRIB: Scribbled homolog (Drosophila) " 388.415192 176.395325 0.052956
                                    ⋮
1416 "MST4: Mst3 and SOK1"  508.448242  403.297 180 0.072090
*Edges
547 549 0.001489
                                    ⋮
```

*Figura 5-3: Ejemplo de fichero BioFactros2K5.net. En la primera línea se encuentra el número de genes que compone la microarray. A continuación se encuentra la información relativa al identificador, nombre, posición (coordenadas en x y coordenadas en y) y color de cada gen y finalmente, separadas mediante la línea *Edges aparecen las aristas del grafo (gen_1 gen_2 correlación que mantienen).*

5.2.9 Ficheros relbyGenF.txt

Finalmente en los ficheros relbyGenF.txt se encuentran las relaciones de expresión no lineales filtradas de la microarray. Cada línea del fichero representa una relación no lineal diferente que se representa como:

$$T{:}g gen_1 g gen_2 h0.75 d0.3\ c$$

Dónde:

- T representa el tipo de relación no lineal.
- gen_1 y gen_2 son los dos que mantienen dicha relación
- c es la correlación existente entre los dos genes.

5.2 Descripción y uso de los programas

A continuación detallo por orden de aparición las propiedades de los diversos programas que forman mi proyecto.

5.2.1 Programa Lanzadora

El programa lanzadora es el código principal de este proyecto. Desde el programa lanzadora se realizan las llamadas a los diversos procesos que componen el tratamiento de las microarrays.

A continuación muestro el conjunto de llamadas que realiza este programa.

lanzadora

Cálculo de las correlaciones entre todos los genes de la microarray y detección de las relaciones de expresión no lineales.

factors2samples

correl

Prim

Comprobador

Clustering

```
Si (la microarray se ha partido) entonces
    Por cada partición i de la microarray x hacer
        Renombrar fichero de la partición x.genes por x_i.genes
        Renombrar fichero de la partición x.factors por x_i.factors
        Renombrar fichero de la partición x.maxcor por x_i.maxcor
        Renombrar fichero de la partición BioMST.net por BioMST_i.net
        Renombrar fichero de la partición x.clusters por x_i.clusters
        Renombrar fichero de la partición BioFactors2K5.net por BioFactors2K5_i.net
        Renombrar ficheros de la partición relbyGen.txt por relbyGen_i.txt
        Renombrar fichero de la partición clusters_normClass.clust por clusters_normClass_i.clust
        Renombrar fichero de la partición clusters_normClass_normGen.clust por clusters_normClass_normGen_i.clust
        Renombrar fichero de la partición clusters_normGen.clust por clusters_normGen_i.clust
        Renombrar fichero de la partición clusters_sinNormalizar_i.clust por clusters_sinNormalizar_i.clust.
    Fin Por
Fin Si

Si (la microarray se ha partido) entonces
    Por cada partición i de la microarray hacer
        grafoclusters(Partición i)
    Fin Por
Sino grafoclusters(microarray)
Fin Si

Si (la microarray se ha partido) entonces
    Por cada partición i de la microarray hacer
        Filtrado_Nolineal(Partición i)
    Fin Por
Sino Filtrado_Nolineal (microarray)
Fin Si
```

5.2.2 Programa Lanzadora: Proceso de cálculo de las correlaciones entre todos los genes de la microarray

Este proceso, calculado dentro del programa principal Lanzadora, proporciona un fichero con la estructura mostrada en la figura 3-2.

Dicho fichero, llamado x.factors proporciona las correlaciones de todos los genes con todos los genes de la microarray x.

5.2.3 Programa Lanzadora: Proceso de detección de las relaciones de expresión no lineales que mantienen los genes

Este proceso, al igual que el anterior, se calcula dentro del programa principal Lanzadora. Mediante este proceso se obtienen todas las relaciones de expresión no lineales de la microarray a estudiar.

Todas las relaciones se guardan en la carpeta nonlineal/ dentro del directo de la microarray. De forma detallada, en esta carpeta se tendrán los siguientes directorios y ficheros:

/class (curvas de baja correlación)
-- *.curv ← ficheros de curvas
-- stats.log ← estadísticas de la clasificación
-- relbyGen.txt ← listado de relaciones entre genes/clases
-- *.class ← clases de curvas que contienen los pares de genes a los que pertenece la clase
/classHeighF (curvas de alta correlación)
-- *.curv ← ficheros de curvas
-- stats.log ← estadísticas de la clasificación
-- relbyGen.txt ← listado de relaciones entre genes/clases
-- *.class ← clases de curvas que contienen los pares de genes a los que pertenece la clase
/clusters <- datos generados por el proceso que se utilizan en otros programas
/clustersSinCERR <- Clusters de genes basados en relaciones de expresión no lineales entre genes
-- clustering.log ← Log del proceso de clustering
-- clusters_normClass.clust ← clustering normalizando clases (columnas)
-- clusters_normClass_normGen.clust ← clustering normalizando clases y genes (columnas y filas)
-- clusters_normGen.clust ← clustering normalizando genes (filas)
-- clusters_sinNormalizar.clust ← clustering sin normalizar
-- *.dat ← datos generados por el proceso
/normal ← datos generados por el programa principal que se utilizan en otros programas
/classRH_F10 ← datos generados por el programa principal que se utilizan en otros programas

5.2.4 Programa factors2samples

Como he comentado anteriormente, el programa factors2samples es el encargado crear un fichero por cada gen de la microarray. Este fichero, llamado *x.txt* contiene los 500 genes mejor correlacionados con el gen x. Todo este conjunto de ficheros se guarda en el directorio factors/ de la carpeta de la microarray a estudiar.

La llamada a este proceso tiene la siguiente sintaxis:

./factors2samples n s -b 500

donde n representa el número de genes y s el número de samples de la microarray..

5.2.5 Programa correl

Este programa, encargado de calcular la mejor correlación de cada gen se invoca mediante la siguiente llamada:

./correl p n ruta

dónde:

- p: es el parámetro de búsqueda.
  - Si p = 1, como es el caso de este proyecto, se buscan las menores correlaciones entre los genes.
  - Si p = 0, el proceso se realiza de forma análoga pero se buscan las mayores correlaciones entre los genes.
- n: es el número de genes de la microarray
- ruta: indica la ruta donde se encuentra el fichero x.factors. Su valor por defecto es ../microarray/mx/x.factors, donde x es el nombre de la microarray a estudiar.

El fichero donde se encuentran las mejores correlaciones de cada gen es guardado inicialmente en el directo desde donde es llamado el proceso con el nombre *millorcorrelacio.txt*. No obstante, una vez finalizada la ejecución de este proceso, el programa principal renombra este fichero llamándolo *x.maxcorr* y lo guarda en el directorio de la microarray x.

5.2.6 Programa Prim

Este programa es el encargado del cálculo del minimun spaning tree de la microarray. La sintaxis de su llamada es la siguiente:

./Prim p n ruta_factors ruta_maxcorr

dónde:

- p: es el parámetro de cálculo
  - Si p = 0, como es el caso de este proyecto, se calcula el minimun spaning tree.
  - Si p = 1, el proceso se realiza de forma análoga pero se busca el maximun spaning tree, es decir, el árbol, las aristas del cual tienen peso máximo.
- n: es el número de genes de la microarray
- ruta_factors: indica la ruta donde se encuentra el fichero x.factors. Su valor por defecto es ../microarray/mx/x.factors, donde x es el nombre de la microarray a estudiar.
- ruta_maxcorr: indica la ruta donde se encuentra el fichero x.maxcorr. Su valor por defecto es ../microarray/mx/x.maxcorr, donde x es el nombre de la microarray a estudiar.

El fichero donde se encuentran el MST resultante es guardado inicialmente en el directo desde donde es llamado el proceso con el nombre *mstprim.txt*. No obstante, una vez finalizada la ejecución de este proceso, el programa principal renombra este fichero llamándolo *BioMST.net* y lo guarda en el directorio de la microarray.

5.2.7 Programa Comprobador

Como he explicado anteriormente, el programa comprobador tiene como finalidad solventar, en caso que haya, los errores del fichero x.maxcorr debidos a correlaciones iguales en un mismo gen. A continuación se detalla la sintaxis de su llamada:

./Comprobador n ruta_maxcorr ruta_mst

donde:

- n: es el número de genes de la microarray
- ruta_maxcorr: indica la ruta donde se encuentra el fichero x.maxcorr. Su valor por defecto es ../microarray/mx/x.maxcorr, donde x es el nombre de la microarray a estudiar.
- ruta_mst: indica la ruta donde se encuentra el fichero BioMST.net. Su valor por defecto es ../microarray/mx/BioMST.net, donde x es el nombre de la microarray a estudiar.

El fichero resultante es guardado en el directorio de la microarray x reemplazando el antiguo fichero x.maxcor.

5.2.8 Programa Clustering

Este programa, encargado de calcular las particiones i/o clusters en que se divide una microarray, se llama de la siguiente forma:

./Clustering n p x ruta_maxcorr ruta_mst ruta_factors

dónde:

- n: es el número de genes de la microarray
- p: parámetro que indica la vuelta actual. Este parámetro, utilizado en caso de que queden clusters de nivel n huérfanos, sirve para indicar el número de vuelta en que nos encontramos. Su valor por defecto es 0. De este modo, si en tras encontrar las particiones de la microarray quedan clusters huérfanos, se volverá a llamar al proceso Clustering y esta vez el parámetro p valdrá 1.
- x: es el nombre de la microarray.
- ruta_maxcorr: indica la ruta donde se encuentra el fichero x.maxcorr. Su valor por defecto es ../microarray/mx/x.maxcorr, donde x es el nombre de la microarray a estudiar.
- ruta_mst: indica la ruta donde se encuentra el fichero BioMST.net. Su valor por defecto es ../microarray/mx/BioMST.net, donde x es el nombre de la microarray a estudiar.
- ruta_factors: indica la ruta donde se encuentra el fichero x.factors. Su valor por defecto es ../microarray/mx/x.factors, donde x es el nombre de la microarray a estudiar.

La salida de este programa depende de su propia ejecución. En caso que la microarray a estudiar sea de tamaño pequeño, es decir, en caso que no sea necesaria su división en particiones, el fichero resultante será x.clusters y se guardará en el directorio de la microarray x.

Por el contrario, es decir, si la microarray a analizar ha sido partida se obtendrán los siguientes ficheros:

- directorio desde donde se lanza el proceso:
  - Genes.txt: Fichero que indica el número de genes de cada partición.
  - Renombrado.txt: Fichero que indica por cada vuelta el número de particiones de microarray que se han creado.

- directorio de la microarray: Por cada vuelta i y por cada partición j creada en dicha vuelta se obtendrán los siguientes ficheros:
  - x.factors_i_j
  - BioMST.net_i_j
  - clusters_i_j.txt
  - x_i_j.genes
  - Conversor_i_j.txt
  - /nonlineal/class/relbyGen_i_j.txt
  - /nonlineal/classHeighF/relbyGen_i_j.txt
  - /nonlineal/classHeighF/clustersSinCERR/clusters_normClass_i_j.clust
  - /nonlineal/classHeighF/clustersSinCERR/clusters_normClass_normGen_i_j.clust
  - /nonlineal/classHeighF/clustersSinCERR/clusters_normGen_i_j.clust
  - /nonlineal/classHeighF/clustersSinCERR/clusters_sinNormalizar_i_j.clust

No obstante, y como se verá posteriormente en el programa Lanzadora, estos ficheros serán renombrados una vez acabe la ejecución del proceso Clustering. El resultado final de estos ficheros será el siguiente:

  - x_y.factors
  - BioMST_y.net
  - x_y.clusters
  - x_y.genes
  - Conversor_y.txt
  - /nonlineal/class/relbyGen_y.txt
  - /nonlineal/classHeighF/relbyGen_y.txt
  - /nonlineal/classHeighF/clustersSinCERR/clusters_normClass_y.clust
  - /nonlineal/classHeighF/clustersSinCERR/clusters_normClass_normGen_y.clust
  - /nonlineal/classHeighF/clustersSinCERR/clusters_normGen_y.clust
  - /nonlineal/classHeighF/clustersSinCERR/clusters_sinNormalizar_y.clust

donde x representa el nombre de la microarray e y el número de la partición.

5.2.9 Programa grafoclusters

Como he comentado anteriormente, este programa es el encargado de generar el layout. La sintaxi de su llamada es la siguiente:

./grafoclusters n ruta_factors ruta_mst ruta_clusters ruta_genes

donde:

- n: es el número de genes de la microarray
- ruta_factors: indica la ruta donde se encuentra el fichero x.factors. Su valor por defecto es ../microarray/mx/x.factors, donde z es el nombre de la microarray a estudiar.
- ruta_mst: indica la ruta donde se encuentra el fichero BioMST.net. Su valor por defecto es ../microarray/mx/BioMST.net, donde x es el nombre de la microarray a estudiar.
- ruta_clusters: indica la ruta donde se encuentra el fichero x.clusters. Su valor por defecto es ../microarray/mx/x.clusters, donde x es el nombre de la microarray a estudiar.
- ruta_genes: indica la ruta donde se encuentra el fichero x.genes. Su valor por defecto es ../microarray/mx/x.genes, donde x es el nombre de la microarray a estudiar.

El fichero donde se encuentra el layout resultante es guardado inicialmente en el directo desde donde es llamado el proceso con el nombre BioFACTORS.net. No obstante, una vez finalizada la ejecución de este proceso, el programa principal renombra este fichero llamándolo *BioFACTOR2K5.net* y lo guarda en el directorio de la microarray.

5.2.10 Programa Filtrado_Nolineal

Como he comentado anteriormente este programa es el encargado de filtrar las relaciones de expresión no lineales que posteriormente se mostrarán en el applet. La sintaxis de su llamada es la siguiente:

./Filtrado_Nolineal ruta_nolineal s

dónde:

- ruta_nolineal: indica la ruta donde se encuentra el directorio nolineal de la microarray a estudiar. Su valor por defecto es ../microarray/mx/nonlineal/, donde x es el nombre de la microarray a estudiar.
- s: parámetro que indica la partición de la microarray que se desea filtrar. En caso que se quieran filtrar las relaciones de expresión no lineales de una microarray pequeña, este parámetro valdrá 0.

Los ficheros resultantes de este proceso, llamados relbyGen_y.txt o relbyGenF.txt, según si se trata una partición y de la microarray o de la microarray completa, se guardan en los directorios /nonlineal/class para el caso de las relaciones de baja correlación y en /nonlineal/classHeighF para el caso de relaciones con correlación alta.

6. Conclusiones

Por último, en esta sección haré una breve valoración del proyecto y del plan de objetivos.

Primeramente me gustaría resaltar que los objetivos marcados para la realización del proyecto han sido alcanzados con creces. Como resultado de mi trabajo ahora se ofrece una nueva herramienta muy útil para los investigadores en el campo de la biología molecular y totalmente adaptada al crecimiento en el volumen de datos que dicha ciencia genera.

A continuación expongo de forma detallada la relación de objetivos que ocupaban mi proyecto y su estado actual.

- Objetivos relacionados con las modificaciones en el preproceso:
  - Con mi proyecto he conseguido la máxima funcionalidad, entendivilidad y operatividad del aplicativo tanto para microarrays pequeñas como para microarrays grandes con todo el rango de tamaños intermedios. He conseguido incluso reducir drásticamente tiempos de espera gracias a las optimizaciones realizadas en el código de partida, hecho muy valorado dada la gran cantidad de datos a procesar.
  - Con las nuevas fórmulas dependientes del número de genes he alcanzado el resultado esperado para la criba de las relaciones de expresión no lineales entre genes. Gracias a esto el estudio de los cambios de fenotipos se ha convertido en una actividad más práctica y amena para los investigadores.
  - El algoritmo implementado para la partición en clusters, hyperclusters o hyperclusters de segundo orden ha resultado de gran utilidad. Gracias a su diseño dinámico no es necesario decidir el nivel de los clusters, el mismo algoritmo decide cómo va a agrupar los genes en función de las características de la microarray (número de genes y correlación de los mismos). Por otra parte, en caso de que en un futuro se requiera trabajar con microarrays de orden mucho mayor (del orden de los 300.000 genes) no sería necesario realizar grandes cambios en el programa. Dado su diseño fácilmente escalable, bastaría con modificar la escala de agrupación para conseguir nuevos niveles.
  - La adaptación del layout para que funcione con microarrays de gran orden se ha realizado satisfactoriamente. Por otra parte, gracias a esta tarea he conseguido localizar y solucionar errores que podían producir fallos en nuevos cálculos. De esta forma, he conseguido que el layout sea un proceso correcto y robusto.
  - He conseguido también realizar la partición del layout, el mínimum spanning tree y demás datos necesarios para el applet satisfactoriamente. Esta tarea se ha realizado de forma que el tiempo de espera sea mínimo y se puedan tratar cada una de las particiones de la microarray como microarrays independientes, pero teniendo en cuenta que forman parte de una microarray gobal.
  - Finalmente, he conseguido mejorar la visualización de las relaciones de expresión no lineales gracias al filtrado por tipología. Por otra parte, con esta operación el applet resulta más robusto además de manejable ya que se impiden fallos por excesiva carga de información.

- Objetivos relacionados con las adaptaciones en el applet:
  - El tratamiento diferenciado de las microarrays pequeñas y de gran tamaño ha sido realizado con éxito. Gracias a esto he conseguido optimizar tiempos de espera ya que en caso de microarrays pequeñas, no son necesarias comprobaciones relativas a segmentos.

    - Las modificaciones realizadas al applet para trabajar con las diferentes particiones de la microarray hacen posible que los tiempos de espera sean mínimos. Las diferencias de tiempo de las diversas particiones respecto a microarrays enteras son inapreciables.
    - La coordinación con las aplicaciones externas al applet y la coordinación entre los distintos applets que contienen los diferentes hyperclusters se ha realizado sin problemas.

- Adaptaciones del aplicativo web:
    - Finalmente, he conseguido que el aplicativo web abra simultáneamente los diferentes applets que muestran las particiones que conforman el total de genes de una microarray analizada. El investigador podrá ver entonces y relacionar los genes que desea estudiar de una manera cómoda y sencilla sin necesidad de preocuparse por si su microarray es demasiado grande o no.

De esta forma he conseguido que la interfaz web realizada proporcione un aplicativo altamente usable, entendible y con una alta operativilidad, lo que facilitará su estudio a los investigadores.

Por lo tanto, todos los objetivos se han cumplido incluso proporcionando algunas funcionalidades extras, de forma que el usuario podrá recibir más información útil para su estudio y de una forma más organizada.

Finalmente me gustaría comentar lo gratificante que ha sido realizar este proyecto, y más teniendo en cuenta que he conseguido una aplicación web que en un futuro esperemos que pueda ayudar a descubrir la causa y remedio de algunas patologías.

7. Bibliografía

1. Barrett T, Suzek TO, Troup DB, Wilhite SE, Ngau WC, Ledoux P, Rudnev D, Lash AE, Fujibuchi W, Edgar R: **NCBI GEO: mining millions of expression profiles – database and tools.** *Nucleic Acids Res* 2005, 33:D562-566.

2. Antonov AV, Tetko IV, Mewes HW: **A systematic approach to infer biological relevance and biases of gene network structures.** *Nucleic Acids Research* 2006, 34(1)**:**e6.

3. Parkinson H, Sarkans U, Shojatalab M, Abeygunawardena N, Contrino S, Coulson R, Farne A, Lara GG, Holloway E, Kapushesky M, *et al*.: **ArrayExpress – a public repository for microarray gene expression data at the EBI.** *Nucleic Acids Res* 2005, 33:D553-555.

4. Burgarella S, Cattaneo D, Pinciroli F, Masseroli M: **MicroGen: a MIAME compliant web system for microarray experiment information and workflow management.** *BMC Bioinformatics* 2005, 6(Suppl 4)**:**S6.

5. Huerta M., Cerdano J., Peña D., Rodriguez A. y Querol E: **PCOPGene-Net: Holistic Characterisation of cellular states from microarray data based on continuous and non-continuous analysis of gene-expression relationships**. *BCM Bioinformatics, 2009.*

6. O'Madadhain J, Fisher D, Smyth P, White S, Boey YB: **Analysis and visualization of network data using JUNG.** *Journal of Statistical Software* 2005, VV**:**1-35.

7. Cedano J, Huerta M, Querol E. **NCR-PCOPGene: An Exploratory Tool for Analysis of Sample-Classes Effect on Gene-Expression Relationships** *Advances in Bioinformatics,* vol. 2008

8. Delicado, P. **Another look at principal curves and surfaces.** *Journal of Multivariate Analysis, 77, 84-116*, 2001.

9. Delicado, P. and Huerta, M**.: 'Principal Curves of Oriented Points: Theoretical and computational improvements'.** *Computational Statistics* 18, 293-315, 2003.

10. Huerta M, Cedano J, Querol E: **Analysis of nonlinear relations between expression profiles by the principal curves of oriented-points approach**. *J Bioinform Comput Biol.* 6:367-386. 2008.

11. Cedano J, Huerta M, Estrada I, Ballllosera F, Conchillo O, Delicado P, Querol E**. A web server for automatic analysis and extraction of relevant biological knowledge.** *Comput Biol Med. 37:1672-1675.*2007.

12. Instituto de Biotecnología y de Biomedicina (IBB) de la Universidad Autónoma de Barcelona. http://ibb.uab.es/ibb

13. Web server for on line microarray analysis supported by the Institute of Biotechnology and Biomedicine of the Autonomous University of Barcelona (IBB-UAB): http://revolutionresearch.uab.es

14. Página Web oficial del National Center for Biotechnology Information (NCBI) que ofrece de manera pública todas sus bases de datos. http://www.ncbi.nlm.nih.gov

Resumen

El IBB ha desarrollado un servidor de aplicaciones: http://revolutionresearch.uab.es para el análisis de microarrays de tamaño reducido. En este servidor podemos encontrar una aplicación web, PCOPGene [5], que permite estudiar y analizar microarrays de tamaño reducido con diversos métodos desarrollados en el IBB.

La tecnología de microarrays, nos proporciona matrices que contienen filas que representan los genes y columnas que representan los experimentos. En cada celda tenemos el nivel de expresión de dicho gen para dicho experimento.

El presente proyecto trata entonces de cómo optimizar y ampliar significativamente las funcionalidades de distintas herramientas de PCOPGene, para poder trabajar con microarrays de gran tamaño. Esto se consigue dividiendo las microarrays originales y trabajando con particiones de éstas.

Resum

El IBB ha desenvolupat un server de aplicacions: http://revolutionresearch.uab.es per l'anàlisi de microarrays de mida reduïda. En aquest server podem trobar una aplicació web, PCOPGene[5], que permet estudiar y analitzar microarrays de mida reduïda amb diversos mètodes desenvolupats al IBB.

La tecnologia de microarrays, ens proporciona matrius que contenen files que representen els gens i columnes que representen els experiments. En cada cel•la tenim el nivell de expressió de cada gen amb cada experiment.

En el present projecte, tracta de com optimitzar i ampliar significativament les funcionalitats de diverses eines de PCOPGene[2], per poder treballar amb microarrays de gran mida. Això s'aconsegueix dividint les microarrays originals i treballant amb particions d'aquestes.

Summary

The IBB center, has developed a application server: http://revolutionresearch.uab.es to analyze the microarrays of reduced size. In this application server we can find a web application, PCOPGene [5], this application allows to study and analyze microarrays of reduced size with many methods developed in the IBB.

The microarrays technology provides us a matrix that contain rows witch represent genes, and columns that are the experiments. In each cells, we obtain the expression of the gene for each experiments.

This project aims to optimize and significantly expand the capabilities of PCOPGene [5], in order to work with large microarrays. This is achieved by dividing the original microarrays and working with partitions of these.