2030 BIOINFORMÁTICA::IDENTIFICAR GENES EN UNA INTERFAZ GRÁFICA VÍA WEB PARA LA COMPARACIÓN DE GENOMAS

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per
Ivan Babitsch Soler
i dirigit per
Jordi Gonzàlez Sabaté
i codirigit per
Mario Huerta
Bellaterra, 9 de Setembre de 2010

Universitat Autònoma de Barcelona

Escola Tècnica Superior d'Enginyeria

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 en Ivan Babitsch Soler.

I per tal que consti firma la present.

Signat:

Bellaterra, 9 de Setembre de 2010

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 en 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, 9 de Setembre de 2010

Tabla de Contenido

1. Introducción

1.1 Motivación de este trabajo

Mi principal motivación fue la oportunidad de actualizar mis conocimientos sobre la tecnología Flash con los últimos avances en el campo, enfrentarme a los problemas de implementación de una interfaz interactiva de alto coste computacional (gran volumen de datos, modelo de eventos, invalidación de regiones de dibujado, etc.…), experimentar frameworks de programación como PureMVC y aproximarme a las nuevas tendencias en soluciones web llamadas RIA (*Rich Internet Applications*). Esto hizo que me implicara con el proyecto desde un primer momento, implicación que fue en aumento por el grado de libertad que me dieron los directores para explotar esta tecnología web al máximo.

Por si esto no fuera suficiente, además, a medida que he ido avanzando en el desarrollo del proyecto he tomado contacto con lo más actual dentro de la línea de investigación en que se enmarca el proyecto: la genómica comparativa, el estudio y detección de rasgos comunes entre los distintos seres vivos, y la secuenciación de nuevos organismos cada día.

La máxima actualidad en lo correspondiente a la secuenciación de genomas (obtención del código genético de un organismo), el altísimo coste computacional que comporta trabajar con estos grandes volúmenes de datos (estamos hablando de cientos de miles de millones de bases) y las limitadas herramientas que disponen los investigadores para este tipo de estudios (debido a su complejidad), llevan a considerar que los esfuerzos dirigidos por el IBB (Institut de Biotecnología i Biomedicina) como punteros y atractivos, al ofrecer nuevas herramientas a la comunidad científica cuyo fruto de investigación revertirá al conjunto de la sociedad. La participación en una de estas iniciativas, esperando que el esfuerzo resulte de utilidad, es altamente gratificante.

1.2 Estado del arte

El genoma de un organismo contiene toda la información genética que define a ese organismo.

Mediante la comparación de genomas se busca la detección de los rasgos comunes que se han mantenido en los organismos vivos. En la historia evolutiva de un organismo algunos rasgos van cambiando con cada salto evolutivo, sin embargo otros rasgos se conservan.

De ahí que sea especialmente útil la comparación entre organismos evolutivamente cercanos, donde las diferencias serán menores, y toda información que se disponga de un organismo puede ser aprovechada para el estudio de sus 'hermanos' evolutivos.

Los organismos vivos se clasifican en tres grandes grupos: Arqueas, Bacterias y Eucariotas.

Cada uno de ellos tiene un rango de longitud de sus genomas diferente, donde los organismos de menor longitud son las Arqueas y las de mayor longitud las Eucariotas (Figura 1). Este gran

volumen de información es el factor más limitativo a la hora de comparar los genomas y también a la hora de visualizar estas comparaciones.

El objetivo de este proyecto es justamente el posibilitar la visualización de esas comparaciones y además hacerlo vía web, de forma que todo el mundo pueda acceder a la comparación de los organismos secuenciados hasta el momento usando internet.

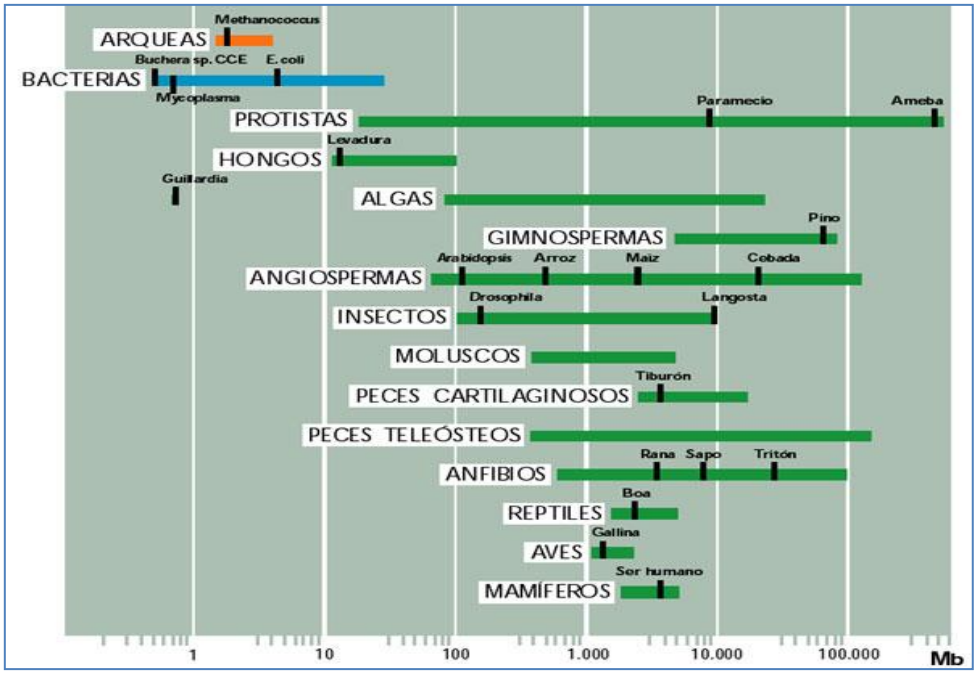

Figura 1: Rangos de los tamaños de los genomas en Millones de bases.
Las tres grandes clasificaciones de los organismos vivos representadas en diferente color: naranja: Arqueas, Azul: Bacterias y Verde: Eucariotas.

A fin de hacer posible la comparación entre genomas es necesario condensar el volumen de datos resultante y mantener la información significativa. Para ello existen diferentes maneras de estructurar la información.

Una de ellas es el uso de MUMs (Maximal Unique Matching, secuencia máxima y única encontrada en ambos genomas) y SuperMUMs (agrupación de las anteriores mediante *Approximate String Matching*).

Mario Huerta (Co-director del proyecto) y Xavier Messeguer desarrollaron una manera eficiente en coste y tiempo el cómputo los MUMs, desarrollando el algoritmo MUMOL[1] que usaremos en este trabajo.

A día de hoy las herramientas existentes para la comparación de genomas completos son:

- MUMmer (1999). Aplicación de escritorio para la alineación de genomas completos. Es el primero en basarse en la búsqueda de MUMs. Solo puede realizar la comparación de dos genomas simultáneamente [2].
- MALGEN (2003). Es el acrónimo de *Multiple ALignment of GENomes* y es una herramienta web para la exploración de relaciones entre secuencias de ADN. Está basada en el cálculo de MUMS y pueden ser calculadas y representadas para más de dos secuencias simultáneamente haciendo uso del algoritmo de *MUMs On-Line* (MUMOL) para encontrar los MUMs.[3]
- M-GCAT (2006). Es el acrónimo de *Multiple Genome Comparison and Alignment Tool* y es una herramienta interactiva de escritorio para la comparación de genomas. Puede realizar comparaciones de varios genomas simultáneamente. Para el cálculo de MUMs se utiliza el MUMOL. [4]
- Mauve (2008). Aplicación de escritorio para la alineación de genomas completos. Permite la comparación múltiple usando multi-MUMs (*Multiple Maximal Unique Matches*, agrupaciones de MUMs).[5]
- CocoNut (2008). Acrónimo de *Computational **C**omparative ge**N**omics **U**tility **T**oolkit*, aplicación multipropósito para escritorio que permite la comparación simultánea de múltiples genomas calculando multi-MEM (*Multiple maximal exact match*, agrupación de MEMs, una variante menos estricta que los MUMs).[6]

Gran parte de estas aplicaciones están limitadas a exploraciones de genomas 'relativamente' pequeños del tipo Arquea, Bacteria o pequeños Eucariota. En la aplicación web que se quiere desarrollar se pretenden comparar desde pequeñas Arquea (cientos de miles de bases) hasta Eucariotas medianas (miles de millones de bases). El alto coste computacional es difícil de evitar, pero al tratarse de un servidor solo necesita computarse la comparación entre dos genomas una sola vez para todos los usuarios que quieran consultar dicha comparación. El reto reside ahora en como mostrar esa comparación vía web cuando estamos comparando miles de millones de bases y la longitud de los genes más grandes rondan las 10000 bases.

Interesa entonces conseguir una aplicación accesible con navegador web (alta accesibilidad), que permita la comparación entre múltiples genomas utilizando estructuras de datos representativos (MUMs y SuperMUMs), que muestre los genes de los genoma involucrado y que pueda operar con cualquier tamaño de genoma con un coste computacional soportable por las máquinas cliente.

El gran volumen de datos a representar (MUMS, SuperMUMs y genes) provoca que simples procesos de ordenación o de filtro de la información a mostrar provoque tiempos de respuesta inaceptables para una interfaz de exploración. Para solventar este hecho se suele generar una gráfica estática que el usuario puede visualizar en mayor o menor detalle, pero ello, además de ser imposible para el volumen de datos tratados, limitaría las opciones de interacción y manipulación de los datos que pueden servir al usuario para obtener la información de interés que éste busque en cada momento. El conjunto de soluciones adoptadas y los resultados de su aplicación se expondrán en detalle en el transcurso de la memoria.

1.3 Objetivos

La estrategia en que se enmarca la solución desarrollada se basa en la subdivisión del problema de la comparación entre genomas en dos partes:

1. Procesamiento de las secuencias de los genomas generando las estructuras de datos relativas a MUMs, SuperMUMs, genes e información estadística necesaria para la interfaz.
2. Representación visual de los datos obtenidos.

Denominamos pre-proceso al punto 1, la generación de datos que consumirán varias aplicaciones web. Estos serán computados una vez y estarán accesibles en el servidor web.

La representación visual de los datos obtenidos será una interfaz web que permitirá la comparación entre los múltiples genomas.

Los objetivos marcados (algunos refinados durante el desarrollo) han consistido en:

- Crear una interfaz web que mediante el uso de MUMs y SuperMUMs permita comparar genomas completos.
- Permitir tanto una visión global de las relaciones (esqueleto) como llegar a un detalle a nivel de bases.
- Permitir que cuando no todo el genoma sea visible se pueda decidir en qué área de la comparación se ubica el área de visualización. Permitir visualizar la comparación entre cualquier parte de un genoma con cualquier parte del otro genoma en el área de visualización para cualquier *zoom*.
- No mostrar información no útil, como MUMs diagonales cuyo punto de origen y destino no estén dentro del área de visualización activa.
- Crear un programa que recupere, procese y ensamble los genes identificados de un genoma, detectando los cromosomas en los que están ubicados.
- Mostrar en la interfaz los genes de los genomas involucrados en cada comparación, permitiendo acceder a una información ampliada de cada gen.
- Permitir la selección de genes de manera individual o múltiple resaltando la correspondencia de este gen con el otro/otros genoma (MUMs y SuperMUMs).
- Aceptar regiones de los genomas por parámetro para resaltar la conservación de ese genoma en el otro/otros genomas (MUMs y SuperMUMs).
- Adaptar la fase de pre-proceso de generación de datos para la comparación de grandes genomas y optimizar el consumo de memoria del cálculo de MUMs para estos grandes genomas.
- Conseguir representar en la interfaz cualquier tipo de genoma (Arquea, Bacteria y Eucariota).
- Permitir en la interfaz la comparación múltiple de 9 genomas simultáneamente.
- Permitir trabajar a nivel de cromosoma, acotando la exploración de un genoma completo a los cromosomas que lo componen.
- Optimizar la interfaz a fin de obtener buenos tiempos de respuesta durante la exploración de las comparaciones.

- Ajuste automático de la información a mostrar en función de la capacidad de cómputo de la máquina cliente.
- Maximizar el tamaño de las gráficas para una mayor accesibilidad a la información representada.
- Diseño líquido, adaptable al tamaño de ventana disponible.
- Permitir trabajar a pantalla completa.

1.4 Organización de la memoria

Como ya se ha comentado, el bloque principal de este trabajo consiste en el diseño e implementación de una nueva interfaz web que permita la exploración en detalle de las relaciones entre múltiples genomas. A continuación explicaremos como esta interfaz ha podido llevarse a cabo.

Primero veremos la sección de fundamentos teóricos, que nos permitirá adquirir las nociones tanto biológicas (genómica) como técnicas (MUMs y SuperMUMs) necesarias para entender la información mostrada por nuestra herramienta.

Tras ella se expone la sección principal de la memoria: Fases de desarrollo y estrategias.

Fases de desarrollo y estrategias nos muestra primero las fases de desarrollo planificadas, se ubica luego al lector en el contexto en que se enmarca la interfaz implementada, y por último se detallan el conjunto de estrategias aplicadas para conseguir los objetivos marcados.

Todas las estrategias aplicadas aparecen ampliamente justificadas. De igual forma, todos los cambios de enfoque requeridos durante el transcurso del proyecto aparecen junto a las causas que los motivaron.

A las fases y estrategias de desarrollo le sigue un breve informe técnico. En este se detallan los parámetros de los programas, el formato de los archivos de entrada y salida, la estructura de directorios del servidor, y otros detalles técnicos que han de facilitar tanto su despliegue como mantenimiento.

Se concluye la memoria con la sección Conclusiones y unas propuestas de mejora que personalmente considero atractivas.

Para estudiar el comportamiento del interfaz y valorar su uso en el 'mundo real' se ha trabajado con genomas reales. Se ha computando el pre-proceso para:

- Todos los genomas del dominio Arquea.
- Selección de un conjunto de Bacterias consideradas de interés. Y se han dejado computando las comparaciones entre el total de bacterias (más de 1000).
- Genomas del Homo Sapiens y Macaca Mulatta (Macaco) como representación del dominio Eucariota.

Todas las figuras mostradas en la memoria representan comparaciones reales entre genomas. La Tabla 1 muestra el volumen de datos implicados en una exploración simple para un par de genomas. Esto puede dar idea del reto al que nos enfrentamos.

Pares de Genomas	Tipo	MUMs	SuperMUMs[1]	Genes
Methanococcus_maripaludis_C6 Methanococcus_maripaludis_C7	Arquea	20.715	2.303	243
Escherichia_coli_APEC_O1 Escherichia_coli_O55_H7_CB9615	Bacteria	52.725	1.976	7.557
Homo_Sapiens Macaca_Mulatta	Eucariota	81M[2]	2M	30.781

Tabla 1: Volumen de datos a representar gráficamente en el interfaz.(M: Millones)
Ejemplos de referencia para cada uno de los tres dominios de genomas existentes , indicando el número de MUMS y SuperMUMs que representan sus relaciones junto al número de genes identificados en cada par de genomas.

Terminología

Denotamos como MUMs a los Maximal Unique Matchings.
Cuando se denota a los SuperMUMs, se referencia tanto a los SuperMUMs como sus MUMs no absorbidos[3].
Denotamos como (S)MUMs cuando se quiere referenciar de manera indistinta tanto a MUMs como a SuperMUMs y sus MUMs no absorbidos.

[1] Aunque denotados como SuperMUMs los números también incluyen los MUMs no absorbidos por los primeros.

[2] En la exploración de grandes genomas se desactiva la visualización de MUMs, permitiendo únicamente la exploración a nivel de SuperMUMs.

[3] Los MUMs no absorbidos son aquellos MUMs que tras el proceso de generación de SuperMUMs no han acabado formando parte de ningún SuperMUM, no han sido absorbidos por un SuperMUM.

2. Fundamentos teóricos

2.1 Conceptos Previos Biológicos

La célula es el elemento de menor tamaño que puede considerarse vivo.
Las características de la célula de un organismo se utilizan para su clasificación, por ejemplo, la existencia de de un orgánulo nuclear (núcleo) en el medio intracelular (citoplasma) es una de las características propias de los Eucariotas, tipo de células de las que el ser humano está formado. Más adelante detallaremos esta clasificación.

Las principales moléculas que constituyen a la célula son: glúcidos, lípidos, proteínas y los ácidos nucleicos.

Glúcidos y lípidos tienen funciones energéticas o estructurales. Los glúcidos más usuales son glucosa, fructosa, lactosa y sacarosa y sus polímeros[4]: glucógeno, almidón o celulosa. Mientras que los lípidos que nos resultaran más familiares son Omega-3, colesterol u hormonas sexuales.

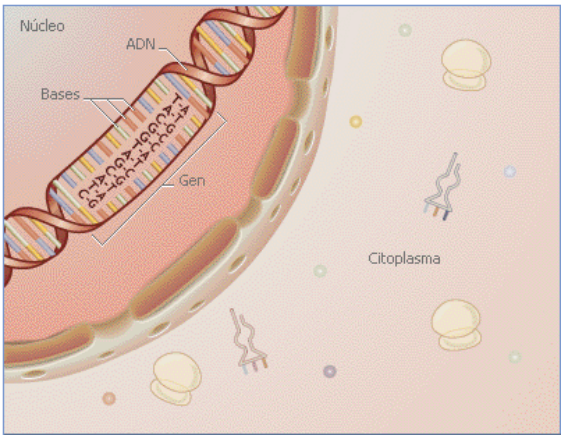

Figura 2: Núcleo y Citoplasma de una célula Eucariota. En el núcleo encontramos el ADN compuesto por las Bases nitrogenadas y sus agrupaciones en Genes.

Las proteínas y ácidos nucleicos son largas moléculas no repetitivas, polímeros, caracterizadas por la misma secuencia de la que se componen. El orden concreto de estas secuencias es propio de cada especie.

Las proteínas son una sucesión de aminoácidos y estos solo aparecen como una de las veinte versiones sobre el total de aminoácidos existentes, es decir, una proteína es secuencia constituida sobre un alfabeto de veinte símbolos, siendo cada símbolo uno de los aminoácidos posibles.

Estas proteínas son las 'máquinas moleculares' que regulan el metabolismo de una célula, teniendo múltiples funciones como estructurales, catalíticas, sensoriales o motrices.
El conjunto de proteínas que regulan el metabolismo de la célula distinguen la especialización de esta en los organismos pluricelulares, determinando el funcionamiento correcto de esta en su contexto.

Los ácidos nucleicos (el ADN, ácido desoxirribonucleico y el ARN, ácido ribonucleico) se componen por una sucesión de nucleótidos, distinguibles por la base nitrogenada que los constituye. En el ADN aparecen cuatro bases nitrogenadas diferentes: A (adenina), T (timina) ,

[4] Los polímeros son macromoléculas (generalmente orgánicas) formadas por la unión de moléculas más pequeñas llamadas monómeros.

G (guanina) y C (citosina), mientras en el ARN encontramos los mismos nucleótidos a excepción del U (uracilo) en substitución de la timina del ADN.

El ADN está formado por dos cadenas helicoidales unidas por las bases nitrogenadas en una relación de complementariedad: A se complementa con T y G con C.

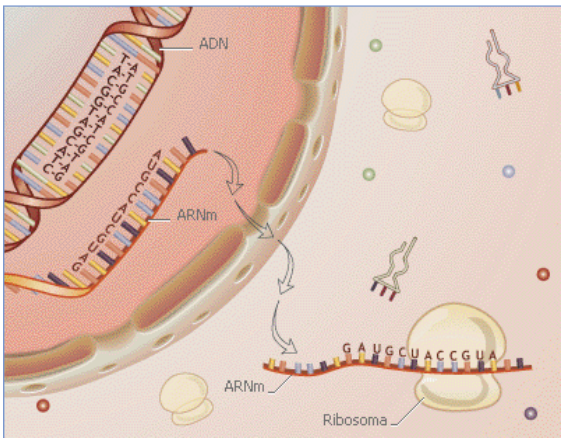

Figura 3: ARNm, ARN mensajero transporta la información extraída de un gen para el ensamblaje de una o múltiples proteínas o ARN

Las funciones del ADN son las de salvaguardar y transmitir la secuencia de los nucleótidos de la que se compone , mientras que los ARN son mucho más versátiles haciendo también desde funciones catalíticas hasta regular la síntesis proteica.

La información codificada en estas bases nitrogenadas son las posibles proteínas o ARNs que se ensamblaran para regular el metabolismo de una célula. Las subsecuencias del ADN que codifican la información de una o múltiples proteínas / ARNs lo denominamos genes.

Existen muchos tipos de ARN, y se suele anotar con un sufijo su función, por ejemplo: ARNm a la molécula de ARN que hace de mensajero, transportando la información de un gen hasta el Ribosoma (orgánulo de la célula encargado de ensamblar las nuevas proteínas).

Podemos observar que el código original del ADN formado por un alfabeto de cuatro símbolos (A,T,G y C) se traducirá en una proteína que a su vez está formada por una secuencia que podríamos codificar con veinte símbolos. De este modo agrupaciones de tres nucleótidos nos permiten codificar a uno de los veinte aminoácidos que sintetizará, con cierta redundancia que podemos interpretar como tolerancia a errores o rastros evolutivos.
A la agrupación de tres nucleótidos del ARNm que dará lugar a un aminoácido se denomina codón.

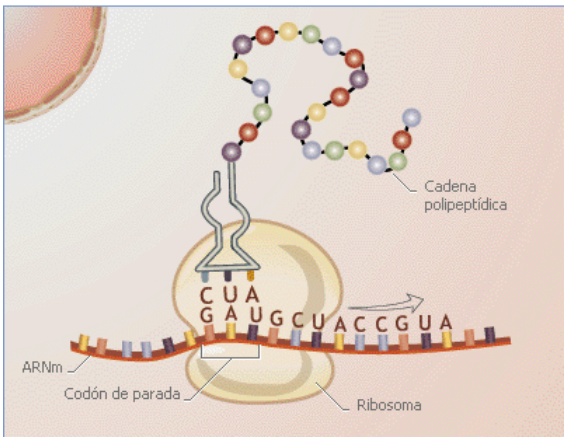

Figura 4: Transcripción del ARNm por el Ribosoma, ensamblando una cadena polipeptídica que dará lugar a la proteína.

El que un gen se exprese o no, es decir, se active su proceso de 'ensamblaje' (lectura de su código desde el ADN hasta la síntesis de la proteína/ ARN que codifica), depende de múltiples factores, como puede ser la existencia de determinadas moléculas en el medio celular que bloquean o desbloquean su posible lectura. La presencia de esta moléculas puede venir de la expresión de otros genes o de el medio extracelular, dando lugar a dinámicas muy complejas.

Denominamos Genoma a la totalidad de información genética que posee un organismo particular codificada en sus moléculas de ADN.

Apreciamos que el hecho de tener identificados los genes dentro del genoma de una especie nos ofrece toda la información necesaria para el estudio del metabolismo de una célula y por ende del organismo en sí.

Remarcar que las partes no codificantes del genoma, conocidos como 'genoma basura' se asociaba a restos evolutivos no funcionales. Hoy en día no se tiene pleno convencimiento de este hecho y se le presuponen que juegan una parte activa en la dinámica celular.

2.2 Evolución biológica

La evolución biológica es el conjunto de transformaciones o cambios a través del tiempo que ha originado la diversidad de formas de vida que existen a partir de un antepasado común.

Actualmente se aceptan los siguientes procesos que afectan a estas transformaciones: selección natural (adaptación al medio), deriva genética (dinámicas propias dentro de una población), mutación (aberraciones del código genético) y flujo genético (migración entre poblaciones).

Considerando la evolución biológica, la comparación de los genomas de distintas especies puede servirnos tanto como indicador de 'proximidad' evolutiva como la detección de rasgos de una especie que se han mantenido. Esto es especialmente útil si tenemos detectados y estudiados los genes de una especie y queremos ver si estos se han mantenido en la de objeto de estudio.

2.3 Árbol Filogenético

La clasificación filogenética es una clasificación científica de las especies basada únicamente en las relaciones de proximidad evolutiva entre las distintas especies, reconstruyendo la historia de su diversificación (filogénesis) desde el origen de la vida en la Tierra hasta la actualidad.

Ideado por Willi Hennig, las relaciones se derivan de las comparaciones de ciertas secuencias de los ácidos nucleicos presentes en las células (observar que quedan excluidos los microorganismos acelulares como los virus, viroides y priones, ya que su inclusión en la categoría de seres vivos tiene una defensa decreciente).

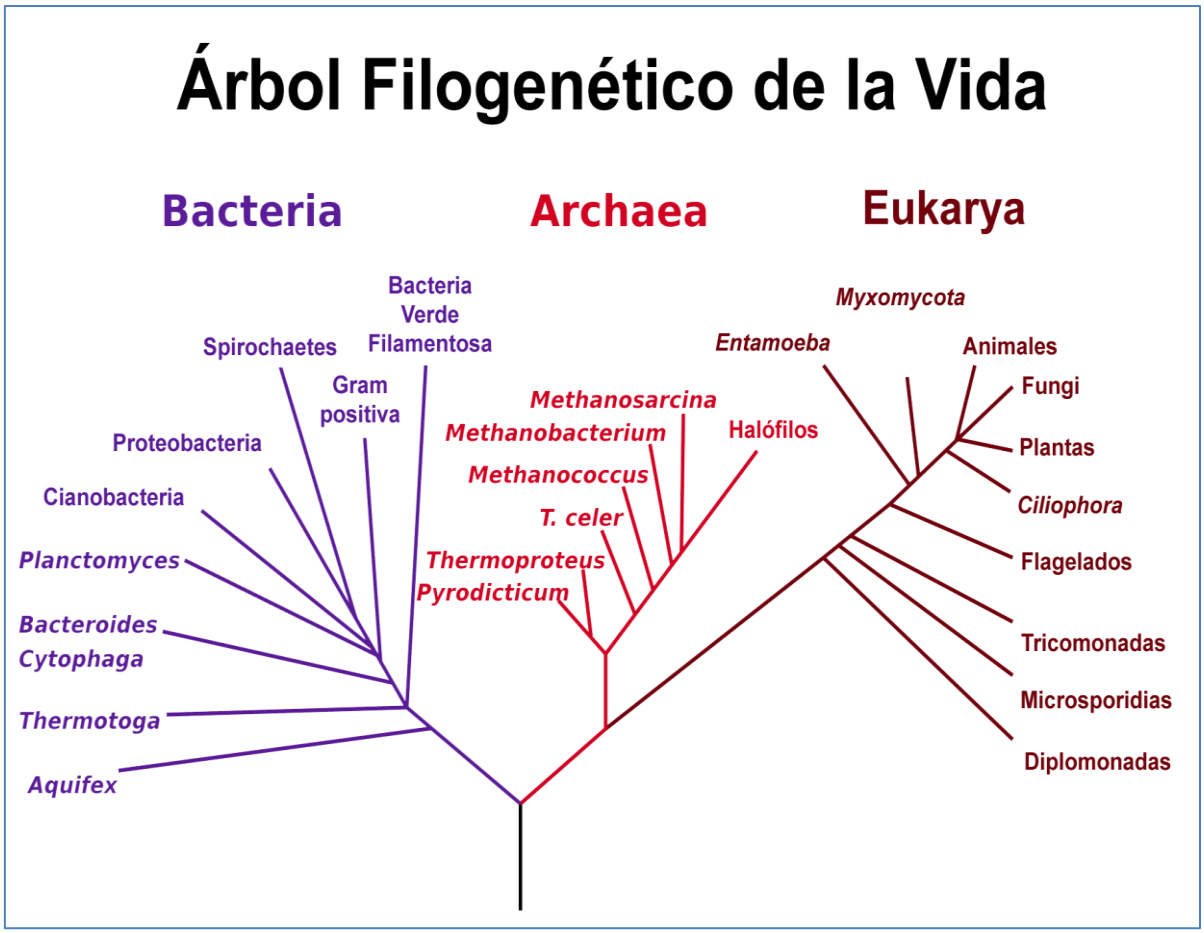

Figura 5: Árbol Filogenético.
Los grandes dominios de la vida: Archaea, Bacteria y Eukarya.

Así pues, los tres grandes dominios de clasificación de las células son: Archaea, Bacteria y Eukarya.

Los dos primeros, Archaea (Arquea) y Bacteria, pertenecen al grupo de Procariotas, células sin núcleo celular diferenciado, es decir, el ADN se encuentra disperso en el citoplasma.
En el dominio Eukarya encontramos las células Eucariotas, donde uno de los rasgos más distintivos es la existencia del núcleo celular que salvaguarda el ADN.

Las células Eucariotas son de mayor complejidad que las Procariotas, de hecho orgánulos como la mitocondria[5] de los Eucariotas (que contienen ADN propio, ADN mitocondrial) son muy probablemente simbiosis de Procariotas precedentes.

A modo de referencia del volumen de datos con que queremos operar, la Tabla 2 se muestran algunos organismos de los tres dominios, tamaño del genoma en millones de bases (Mb), número de genes y la relación de estos (genes/Mb).

[5] La Mitocondria regula la síntesis de ATP, molécula principal utilizada como 'combustible' del metabolismo celular.

Organismo		Tamaño del genoma (Mb)	Número de genes	Densidad génica (genes/Mb)
Nombre común o clase	Nombre científico			
Levadura del pan	*Saccharomyces cerevisiae*	12	6.241	480
Nematodo	*Caenorhabditis elegans*	97	18.424	190
Crucífera	*Arabidopsis thaliana*	125	25.498	204
Mosca del vinagre	*Drosophila melanogaster*	180	13.601	75
Pez globo	*Fugu rubripes*	400	35.000	100
Arroz	*Oryza sativa*	450		
Erizó de mar	*Strongylocentrotus purpuratus*	900	27.350	30
Maiz	*Zea mays*	2.400		
Ser humano	*Homo sapiens*	3.400	35.000	10
Cebolla	*Allium cep*	18.000		
Ameba	*Amoeba dubia*	686.000		
Crenarchaeota	*Aeropyrum pernix*	1,55	1.522	981
Euryarchaeota	*Methanococcus jannaschii*	1,66	1.715	1033
Euryarchaeota	*Archaeoglobus fulgidus*	2,18	2.420	1110
Proteobacteria	*Buchnera sp.* CCE	0,45		
Gram positiva	*Mycoplama genitalium*	0,58	479	831
Proteobacteria	*Buchnera sp.* APS	0,64	564	881
Gram negativa	*Haemophilus influenzae*	1,8	1.727	959
Cianobacteria	*Synechocystis sp.*	3,6	3.168	880
Gram positiva	*Bacillus subtilis*	4,2	4.100	976
Proteobacteria	*Escherichia coli*	4,6	4.288	932

Tabla 2: Tamaño del genoma, número de genes y densidad génica

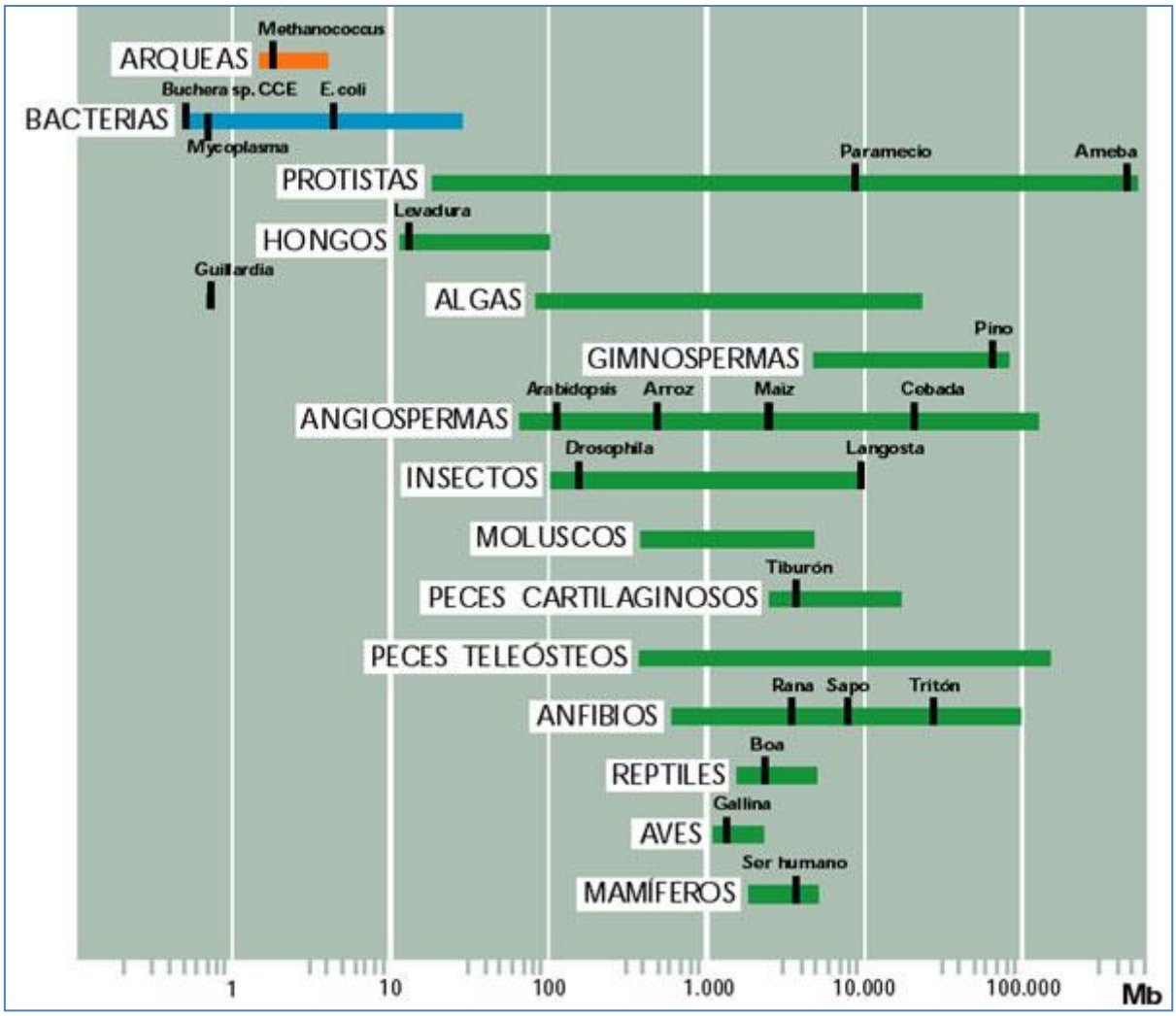

Figura 6: Rango de tamaños de genomas encontrados en los tres dominios de la vida: bacterias (azul), arqueas (naranja) y eucariotas (verde).

2.4 Ramas de estudio

La Genética es la ciencia de la herencia y la biológica variación de los seres vivos, una disciplina de la biología.

Se denomina **genómica** al conjunto de ciencias y técnicas dedicadas al estudio integral del funcionamiento, el contenido, la evolución y el origen de los genomas. Es una de las áreas más vanguardistas de la Biología.

La genómica usa conocimientos derivados de distintas ciencias como son: biología molecular, bioquímica, informática, estadística, matemáticas, física, etc.

El contexto de estudio de este trabajo se enmarca en el área de genómica comparativa, donde mediante la comparación de múltiples genomas se busca encontrar información tanto de los procesos evolutivos como de los rasgos que entre estos se hayan mantenido/perdido.

2.5 Bioinformática

La bioinformática es la aplicación de tecnología de computadores a la gestión y análisis de datos biológicos. En él se aplican campos de estudios interdisciplinarios muy vinculados, que requieren el uso o el desarrollo de diferentes técnicas que incluyen informática, matemática aplicada, estadística, ciencias de la computación, inteligencia artificial, química y bioquímica.

Como hemos visto en la Tabla 2 y Figura 6, el volumen de información que contiene un genoma es inabarcable para las capacidades del ser humano. Por ello, el automatizar procesos de tratamiento de datos buscando estructuras/patrones o determinadas propiedades de un genoma, así como información estadística, se convierte en un necesidad ineludible para afrontar con éxito su estudio.

En nuestro caso, la estructuras de datos extraídas de los genomas (entre pares de genomas) que nos servirán como objeto de estudio son los MUMs.

2.5.1 MUMs

Un MUM (*Maximal Unique Matching*), es la sub-secuencia única máxima coincidente entre dos genomas. En otras palabras, el fragmento de bases correlativo más largo coincidente en dos genomas que no se repite (único).

A parte de extraer y agrupar la información que nos es útil en unidades de información más pequeñas y por tanto más tratables que las bases de un genoma, el conjunto de MUMs conforman lo que se denomina *skeleton* (esqueleto), estructura básica de comparación de genomas que tiene significado biológico.

Como podemos intuir, en la comparación entre dos especies evolutivamente cercanas (que probablemente mantendrán gran parte de su genoma similar al de su antecesor común), obtendremos gran número de MUMS o estos serán de gran tamaño.

MUMs de gran tamaño nos reflejan grandes fragmentos de ADN que se han mantenido intactos en ambos genomas, y por tanto la información que está contiene se expresará de igual manera en ambos organismos (no es exacto en tanto la expresión de los genes viene condicionada por muchos factores).

La condición de que estas subsecuencias sean únicas, es decir, que aparezcan una única vez en ambos genomas, tiene sentido tanto para descartar fragmentos comunes repetidos que no nos ofrecen información biológica, como la de caracterizar la relación entre dos genomas, ya que en caso de existir como única sub-secuencia es un fuerte indicador de herencia común.

Una sub-secuencia del genoma, que por ejemplo, sea un gen que codifique una proteína, no tiene un sentido de lectura preferente, puede estar codificado en sentido izquierda-derecha como derecha-izquierda respecto a la secuencia de genoma.

De hecho, durante el proceso evolutivo puede darse el caso que fragmentos de un genoma se inviertan de una especie respecto a otra, aun codificando la misma información.

Por ello, al procesar los MUMs se consideran dichos fenómenos, se detectan e identifican dando lugar a la clasificación de MUMs directos y MUMs inversos, reflejando las sub-secuencias que se han mantenido el mismo orden o se han invertido.

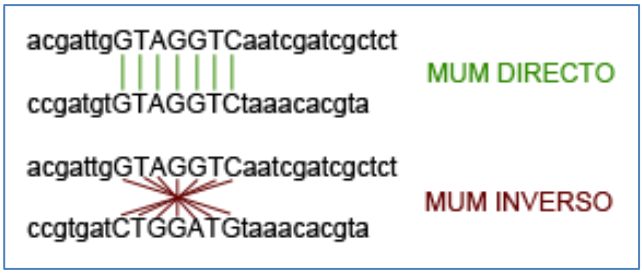

Figura 7: MUM Directo y MUM Inverso
Los MUMs directos mantienen la misma ordenación mientras los inversos el sentido de la secuencia se ha invertido.

2.5.2 SuperMUMs

Para genomas grandes, es útil agrupar los MUMs obtenidos si estos cumplen ciertas condiciones.

La idea es que largas secuencias pueden ser coincidentes en ambos genomas a excepción de unas pocas bases que se han alterado. Ello provocaría que no detectemos un MUM de gran tamaño en este caso, sino múltiples MUMs próximos de menor tamaño.

Dado el polimorfismo y la redundancia del código genético es muy probable que los rasgos codificados en esta secuencia sean los mismos en ambos genomas.

La construcción de un SuperMUM ha de cumplir las siguientes condiciones en ambos genomas simultáneamente:

- gap < (longitud_mum1 + longitud_mum2) * constante.
  El *gap* es la separación en bases que existe entre dos MUMs. Si el *gap* es menor que la expresión ambos MUMs se incluyen en el SuperMUM.
  La constante de la expresión es paramétrica reflejando mayor o menor tolerancia entre las separaciones de los MUMs. Su valor por defecto es dos.
- Dos SuperMUMs con una superposición mayor del 30% respecto a sus respectivas longitudes se fusionan.
- MUMs que en principio no forman parte de un SuperMUM pero quedan inmersos en este último quedan absorbidos, formando parte del SuperMUM.

Observar que las características de los SuperMUMs pueden originar superposiciones, mientras que los MUMs, por definición, nunca se superponen.

3. Fases de Desarrollo y Estrategias

Tras el desarrollo de varios prototipos para evaluar las aproximaciones al problema, se siguió un ciclo iterativo de análisis, implementación y pruebas de cada nueva funcionalidad.

En paralelo al desarrollo de la interfaz, se ha ido profundizando en las características de la tecnología utilizada, ampliando las bases teóricas biológicas y aprendiendo las propiedades de los modelos de datos utilizados. También se han realizado pequeños ajustes al pre-proceso de datos utilizados por la interfaz y se han desarrollado herramientas complementarias a este.

	Fases de desarrollo	Intervalos	Duración
	Análisis viabilidad. Desarrollo de prototipos	Setiembre, Diciembre	0.2 meses 1 mes
	Representación gráfica de MUMs y herramientas de navegación. Adaptaciones del pre-proceso de datos.	Enero → Abril	4 meses
	Herramienta pre-proceso para extracción de información necesaria de los Genes. Representación de Genes.	Mayo → Junio	2 meses
	Representación de SuperMUMs. Selección de fuente de datos y modo gráfico Auto.	Julio	1 mes
	Optimizaciones para Eucariotas.	Agosto	1 mes

Tabla 3: Fases y líneas temporales de desarrollo

Ha habido muchos replanteamientos y cambios de estrategias a lo largo del desarrollo, por lo que se justificará el uso de cada una y de sus motivaciones.

A fin de simplificar su uso como referencia se han agrupado las soluciones planteadas por ámbitos de incidencia.

Empezaremos con una breve introducción al contexto en que se enmarca esta aplicación.

3.1 Contexto de la aplicación

El IBB está desarrollando un servidor de aplicaciones web (accesibles desde un navegador web) para el estudio de la conservación de genes mediante la genómica comparativa. Este servidor consta de 3 partes principalmente. La segunda es la que he desarrollado yo.

3.1.1 Mummy-tree: Árbol filogenético

La primera herramienta a la que se accede es el Mummy-tree. Una aplicación web que muestra gráficamente las relaciones de proximidad evolutiva entre distintos genomas de un mismo dominio (Arquea, Bacteria o Eucariota).

Las relaciones de proximidad se obtienen de datos derivados del cálculo de los MUMs y SMUMS entre cada par de genomas (su grado de correlación), y se representan construyendo un árbol filogenético mediante un *Minimum Spanning Tree*[6].

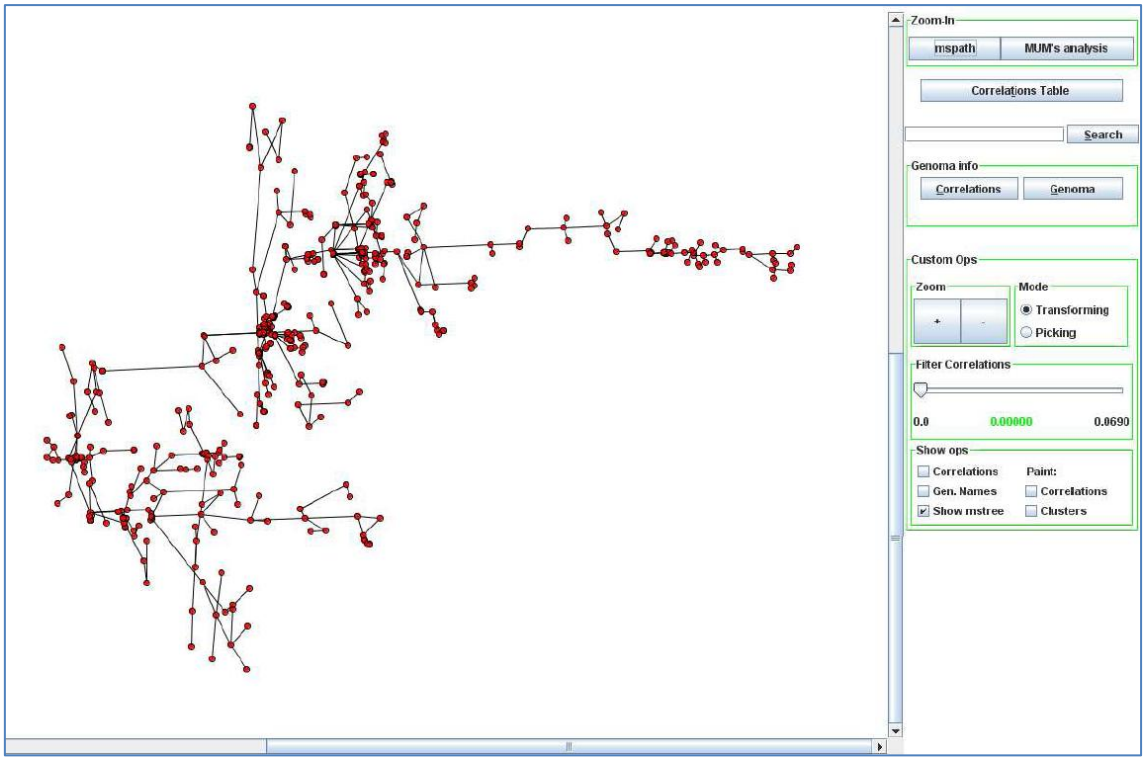

Figura 8: Árbol filogenético representando sus proximidades evolutivas.
Cada vértice representa a un genoma y las aristas la correlación(peso) que con otro genoma de tal manera que se mantenga el grafo conexo y este sea el de menor peso total posible.

Desde esta herramienta el investigador puede seleccionar los genomas de interés sobre los que quiere realizar una comparación detallada, así como los genomas intermedios a estos en términos evolutivos. Con esta acción se invoca la aplicación Mummy que representará gráficamente estas relaciones a nivel de MUMs y SuperMUMs.

3.1.2 Mummy

Mummy es la aplicación web desarrollada por mí en este trabajo. Mummy es una herramienta que permite la exploración en detalle de un conjunto de comparaciones entre genomas.

Mummy muestra gráficamente las relaciones entre múltiples genomas, representando en múltiples gráficos (PairWise) las relaciones en (S)MUMs entre cada par de genomas. Asimismo también muestra los genes de los genomas involucrados.

[6] Minimum Spanning Tree es el subgrafo (árbol) de menor peso (asociado a cada arista, en nuestro caso su grado de correlación) que contiene a todos los vértices (genomas) del grafo original manteniéndose conexo.

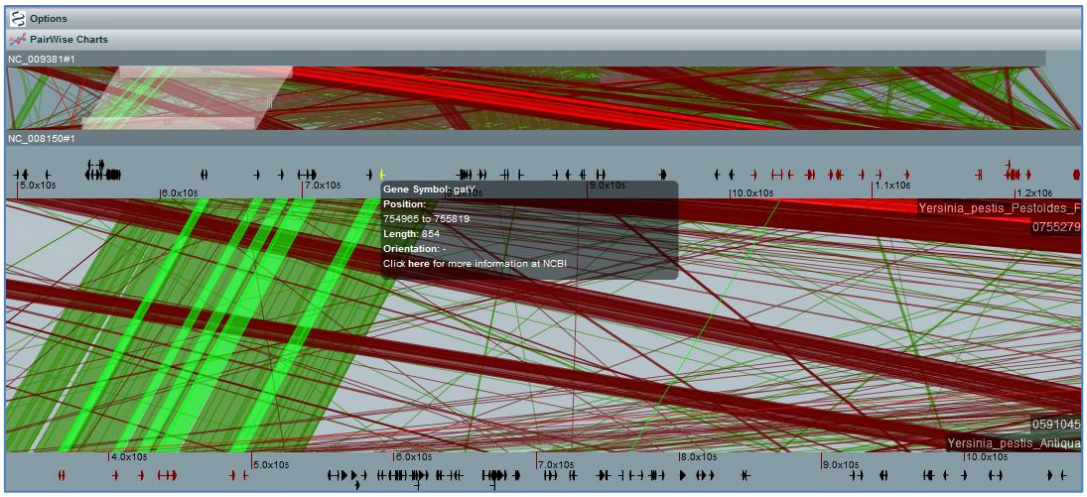

Figura 9: Representación gráfica de MUMs, SuperMUMs y genes de dos genomas de bacteria. Comparación entre las Bacterias Yersinia_pestis_Pestoides_F y Yersinia_pestis_Antiqua

Esta exploración visual ha de permitir a los investigadores (potenciales usuarios) estudiar los procesos evolutivos entre distintos organismos, así como seguir en detalle las variaciones de sus rasgos (genes) a lo largo del proceso evolutivo.

Se trata de aplicación Web desarrollada en Adobe Flex (tecnología hermana de Adobe Flash para la creación de aplicaciones R.I.A., *Rich Internet Applications*).

3.1.3 Pre-Proceso: Generación de datos

Tanto la aplicación de Mummy-tree como Mummy se alimentan de datos previamente computados. A su generación la denominamos fase de pre-proceso.

En este pre-proceso se descargan los datos públicos ofrecidos por el FTP del NCBI [7] y se procesan para generar los datos necesarios: MUMs, SuperMUMs, genes e información estadística.

Fue necesaria la adaptación del programa de cálculo de MUMs (mumol) para poder trabajar con grandes genomas. En grandes eucariotas la secuenciación de un genoma no tiene por qué estar totalmente completada (todavía quedan fragmentos por codificar). Para tratar estos huecos se implementó un pre y post-proceso donde se detectan y corrigen.

Se implementó un programa de descarga, proceso y ensamblaje de los genes identificados para cada genoma, agrupándolos por el cromosoma del genoma en que se ubiquen.

También se creó un pequeño script para el cálculo de los indicies de los archivos de SuperMUMs indicando su ordenación por distintos criterios, a fin de aligerar los procesos de la interfaz (más detalles de la optimización en la sección 3.2.4.2). De esta forma se genera un

[7] National Center for Biotechnology Information

formato de SuperMUMs ampliado identificado por ubicarse en el subdirectorio smumsort de la raíz principal de datos.

Los ficheros necesarios para el correcto funcionamiento de la interfaz Mummy se detallan en la sección 4.1 del Informe técnico.

3.2 Estrategias y Soluciones

3.2.1 Selección de Tecnología y FrameWork de desarrollo

Hoy en día, las principales soluciones tecnológicas para la creación de aplicación web (integrables en un navegador/*browser* web), enmarcadas como aplicaciones R.I.A (*Rich Internet Applications*) que permiten el desarrollo de interfaces complejas con características y capacidades equiparables a las aplicaciones de escritorio son:

- JavaFX, de Oracle Corporation. Conjunto de tecnologías basadas en Java.
- Flex/Flash de Adobe Systems Inc. Basado en Action Script 3 (AS3,ECMAScript Edition 3,ECMA-262) en sus últimas versiones.
- Silverlight de Microsoft. Basado en la plataforma .NET.
- HTML versión 5 junto CSS3 y javascript.

Tanto JavaFX, Flex/Flash como Silverlight se ejecutan es sus respectivas máquinas virtuales: *Java Virtual Machine* (JVM), *ActionScript Virtual Machine* (AVM2) y *Common Language Runtime* (CLR).

JavaFX arrastra una mala herencia de los applets Java (mini aplicaciones integrables en los navegadores), con fama de pesados y pobre rendimiento grafico. Añadido a guerras comerciales (principalmente Microsoft) que dificultaban su activación o instalación de las máquinas virtuales necesarias para su funcionamiento.

Silverlight, muy parecido en desarrollo al Flex de Adobe tiene muy poca cuota de penetración de mercado, motivado principalmente a que existe únicamente para plataformas con Sistema Operativo Microsoft. Iniciativas comunitarias como Moonlight, que son adaptaciones a sistemas Unix/Linux de Silverlight todavía no han conseguido una compatibilidad plena.

Las especificaciones de HTML5, reguladas por el organismo W3C, todavía no son finales (2010). Aunque las últimas versiones de browsers como Chrome o Firefox ya implementan algunas de sus especificaciones todavía es una tecnología 'verde ', poco madura, para poder evaluarla como alternativa tecnológica.

Hoy por hoy, y durante el desarrollo de este trabajo (2009-2010) la tecnología con mayor penetración de mercado, con buen rendimiento grafico y buenas expectativas de mejora en rendimiento con la versión de su máquina virtual (versión 10.1) y publicación de la herramienta de desarrollo Flex 4 (Marzo 2010), es Adobe Flash.

Worldwide Ubiquity of Adobe Flash Player by Version - June 2010		
Flash Player 8 & below	Flash Player 9	Flash Player 10

Mature Markets	99.3%	99.2%	97.5%
US/Canada	99.1%	99.1%	97.5%
Europe	99.3%	99.0%	97.9%
Japan	99.7%	99.7%	97.1%
Australia/New Zealand	99.7%	99.7%	96.8%
Emerging Markets	99.0%	98.9%	96.1%

Tabla 4: Presencia de Flash Player

Tanto por los aspectos mencionados, como el hecho que la versión previa del interfaz ya estaba basada en la tecnología Flash, nos decantamos por esta tecnología para implementar la interfaz.

3.2.1.1 Límites del Flash Player

Límites de Memoria

No existen versiones de 64 bits del Flash Player, únicamente de 32 bits, lo que nos lleva a especular que espacio de direcciones que utiliza es también de 32 bits, limitando las aplicaciones a un tamaño máximo de 4 Gigas de RAM.
Ni la documentación, ni *technotes* reflejan estos límites de memoria, indicando que los límites de memoria están impuestos por el Sistema Operativo.

De hecho, los límites de uso de memoria pueden venir acotados por los navegadores web, el espacio de memoria que reservan para los complementos (*pluggins*), y estos dependen de cada tipo de navegador, versión y Sistema Operativo (S.O.) huésped.

Otras limitaciones que nos afectan al desarrollo se muestran en la Tabla 5, en especial el número máximo de instancias de símbolos (objetos representados gráficamente) limitado a 16.000. En la sección 3.2.3.5 se detalla cómo evitar esta limitación.

Timeline	
	16,000 frames: Exceeding this limit causes the movie playback to stop. While this limit is rarely reached by most developers, it is possible. If your movie must have more than this number of frames,try creating multiple movies with fewer than 16,000 frames each and then linking the movies together using a method such as the ActionScript 2 `loadMovie()` command.
	16,000 layers: Flash is not capable of working with more than 16,000 layers in a movie.
	16,000 loaded movies: You cannot load more than 16,000 movies into Flash player.
	16,000 symbol instances: Flash does not allow more than 16,000 symbol instances in a Flash movie
	2880 x 2880 px canvas size: A Flash movie cannot be larger than 2880 px wide or 2880 px tall.
ActionScript	
	32 KB: This is the limit in file size for any single ActionScript script such as a class. If you require a larger file, try breaking up your code into smaller parts or delegating responsibilities to other classes.

	12 bytes: This is the minimum space in memory taken up by ActionScript variables. This does not include additional space for name or enumeration flags. The actual content (for example, the actual string assigned to a string data type) adds additional bytes.
	15 seconds: This is the amount of time a loop will run until a dialog with the message "A script in this movie is causing Flash Player to run slowly" appears, allowing the user to abort the script.
	125 components: We recommend that Flash Applications built with Version 2 components be limited to 125 components. This is a suggestion for best performance, not a hard rule.

Tabla 5: Límites impuestos por el Flash Player

Límites de tiempo de ejecución

El límite de 15 segundos de ejecución de un bucle que vemos en la Tabla 5 no es exacto, ya que en verdad es el tiempo máximo que el gestor de eventos asigna al método que procese un evento.

Existen directivas que nos permiten ampliar dicho tiempo límite a 60 segundos, pero aun así ello no nos asegura que se pueda generar el evento de control de la máquina virtual.

Para asegurar una buena respuesta de la interfaz se tendido durante su desarrollo, y a medida que se ampliaba el volumen de datos con los que operar, a calcular al inicio de la aplicación todos los datos necesarios con los que se trabaja, a coste de un mayor consumo de memoria. Los puntos críticos en coste de tiempo/cómputo para preparar los datos de la aplicación son:

- Carga de ficheros con los datos a procesar.
- Análisis sintáctico (parseo) de los datos de entra.
- Ordenación de los datos por múltiples criterios.
- Cálculo de las relaciones entre (S)MUMs con los genes.

La carga de datos propiamente dicha, el AS3 nos ofrece primitivas asíncronas que solventan los límites de tiempo.
En cambio, tanto el parseo, ordenación y calculo de relaciones (mapeado) entre (S)MUMs y genes son procesos muy costosos que en ordenadores de poca potencia y/o grandes volúmenes de datos pueden exceder de los límites permitidos.

Llamadas Asíncronas

A fin de minimizar el problema los bucles que operan en los procesos mencionados se han implementado mediante llamadas asíncronas que procesan un número limitado elementos. Mediante el control de variables de estado y llamadas de métodos relegados en el tiempo evitamos el control de tiempo límite impuesto por el gestor de eventos, a la vez que simulamos el flujo normal de carga y proceso de datos que necesitamos para la aplicación.

Esta solución se considera parcial ya que se ha ideado una alternativa mucho más eficiente que se detallará en el apartado de mejoras, principalmente consistente en:

- Los procesos de mapeado de genes, ordenación de *arrays*, etc... han de venir previamente calculados. Dejando a la interfaz la única tarea de representar la información visualmente.

- Optimizar la carga de datos para que la interfaz reciba los datos en formato nativo AS3 (serializándolos mediante AMF3, *Action Message Format*), evitando necesidades de parseo y optimizando al máximo los volúmenes de transferencia de datos.

3.2.1.2 FrameWork de desarrollo

El desarrollo utilizando Flex, programación Orientada a Objetos, nos permite una mejor estructuración del código que su herramienta hermana Flash.

A fin de desacoplar al máximo el modelo de datos, gestión de eventos y representación gráfica se ha aplicado el patrón de diseño Modelo-Vista-Controlador (MVC) de manera no estricta usando el Framework PureMVC para AS3.

No se ha aplicado de manera estricta este patrón en las clases críticas de la aplicación donde se requiere maximizar los tiempos de respuesta. En estas no interesaba delegar a la gestión de eventos la cola de procesos a tratar, ya que se suele sobrecargar la capacidad de cómputo de la máquina cliente y nos interesaba tener un control absoluto de flujo de ejecución (AVM2 trabaja en un único hilo de ejecución y una vez obtenido el 'foco' de ejecución no se desea procesar ningún otro evento hasta completar el proceso crítico)

De este modo se pretende facilitar el mantenimiento de la aplicación y que cualquier cambio de estrategias de implementación tenga un impacto mínimo en el resto del código.

PureMVC

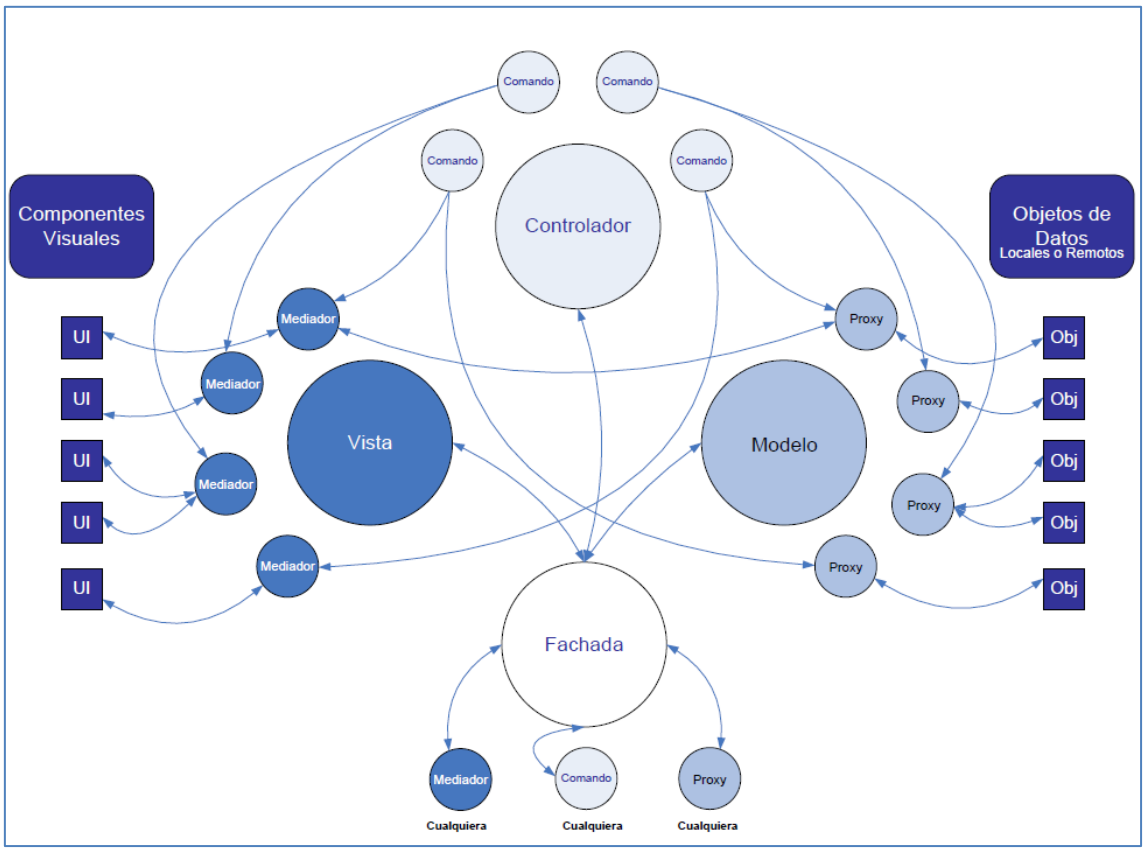

Figura 10: Diagrama Conceptual del FrameWork PureMVC.
Basado en el patrón Modelo-Vista-Controlador que nos permiten desacoplar los distintos componentes de la aplicación.

En esta implementación del clásico meta - patrón de diseño MVC, estos tres niveles de la aplicación se rigen por tres Singleton (una clase de la que sólo una instancia se puede crear) llamados simplemente Modelo, Vista y Controlador. Juntos, ellos se conocen como los "actores principales".

Un cuarto Singleton, la fachada (*facade*) simplifica el desarrollo, proporcionando una única interfaz para la comunicación con los actores principales.

Modelos y Proxies

El Modelo simplemente cachea referencias con nombres a los Proxies. El código Proxy manipula el modelo de datos, comunicando con los servicios remotos si necesita hacerlos persistentes o recuperarlos.

Vistas y Mediadores

La Vista primeramente cachea las referencias con nombres a los Mediadores. El código Mediador administra los Componentes Visuales, agregando oyentes (*listeners*) de eventos, enviando y recibiendo notificaciones hacia y desde el resto del sistema en su nombre y directamente manipulando su estado.

Controlador y Comandos

El Controlador mantiene mapeos con nombres a clases de comandos, el mismo es "*stateless*" (no guarda estados), y sólo se crea cuando se necesita.

Los Comandos deben recibir e interactuar con los Proxies, enviar Notificaciones, ejecutar otros Comandos, y a menudo son utilizados para organizar actividades complejas o sistemas amplios tales como inicios y cierres de aplicación. Ellos son el lugar de residencia de la Lógica de Negocio de su aplicación.

Fachada y principales

La Fachada (*facade*), otro Singleton, inicializa los "actores principales" (Modelo, Vista y Controlador), y provee un único espacio de acceso a todos sus métodos públicos.

Extendiendo de la Fachada, la aplicación obtiene todos los beneficios de los "actores principales" sin tener que importar y trabajar con ellos directamente.

Herramienta de desarrollo: Flash Builder

Una de las herramientas de desarrollo de Adobe Flash es Flex Builder (renombrado en su última versión como Flash Builder), que a diferencia de la herramienta más popular "Adobe Flash" enfocada diseñadores manipulado visualmente formas vectoriales, está orientada a desarrolladores, orientando su programación a objetos (Orientado a Objetos) en el lenguaje AS3.
Flash Builder, derivada de la herramienta Eclipse, es un producto comercial, pero el SDK (*Software Development Kit*) que utiliza esta liberada como código abierto.

Usar esta herramienta de desarrollo nos permite tanto usar patrones de diseño como MVC (Modelo Vista Controlador) como modos de ejecución de depurado o monitorización de recursos.

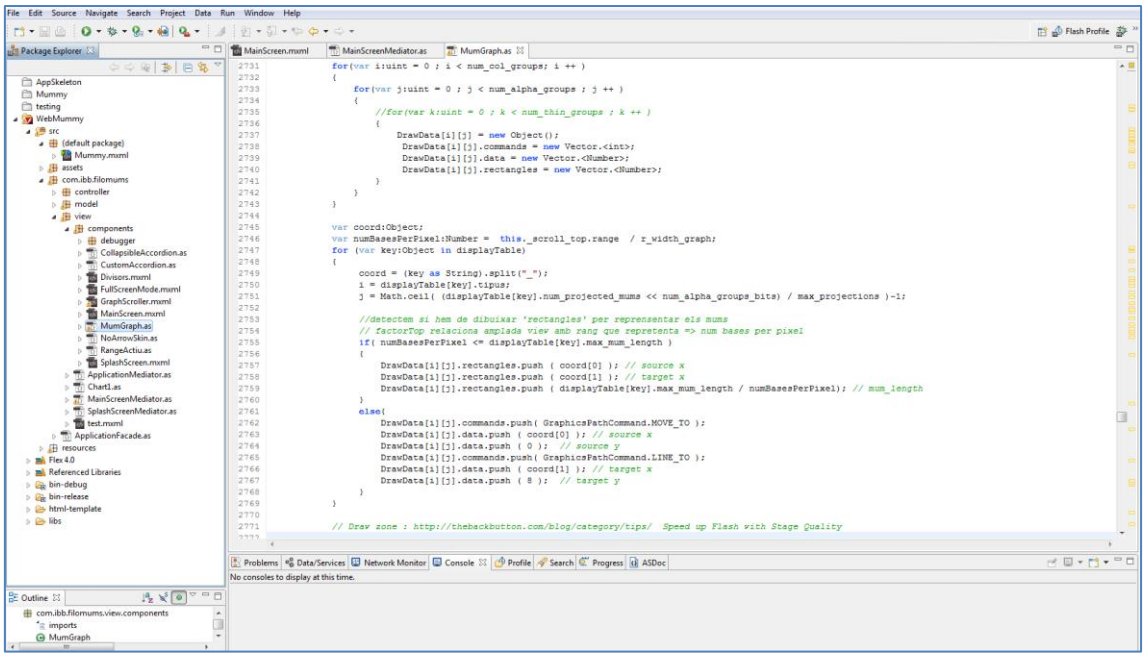

Figura 11: Flash Builder 4, entorno de desarrollo

Desde la versión Flash Player 10 se han introducido nuevas métodos de dibujado, tipos de datos (por ejemplo tipo Vector, que es una colección de datos que mejora hasta un 400% las velocidades de acceso a memoria) y mejor aprovechamiento de la GPU (unidad de proceso gráfico) . Aunque en el aprovechamiento de la GPU, al final se ha reducido a decodificación de video y operaciones especificas de tratamiento de texturas 3D (*shaders*) , que no ha mejorado los procesos de transformación 2D (vectoriales) que utilizamos en nuestra aplicación.

3.2.2 Diseño y usabilidad de la Interfaz

La interfaz se divide en dos espacios de trabajo: Opciones y Visualización / Exploración. El usuario puede cambiar de espacio de trabajo seleccionado las barras que representan cada uno (comportamiento a modo de 'acordeón')

La lista de directivas a seguir en el diseño de la interfaz es la siguiente:

- Maximizar el espacio de representación de las gráficas (PairWise que representan los (S)MUMs que relacionan dos genomas). Ya que es el espacio de trabajo con que el investigador trabajará y estas contienen gran volumen de información.
- Diseño líquido, es decir, que se adaptase automáticamente al espacio de ventana que el entorno (en nuestro caso el navegador web) le asignase.
- Opciones fuesen auto-explicativas, reforzando la ayuda con el uso de *ToolTips* (pequeñas ventanas emergentes que describen la funcionalidad de cada elemento con que el usuario puede interactuar, aparecen cuando el ratón se ubica sobre dicho elemento).
- Limitados por el Flash Player, que reserva el uso del botón derecho para opciones propias de identificación y configuración, y en aras de maximizar la accesibilidad desde

cualquier tipo dispositivo, toda operación de la interfaz tenía que ser accesible desde el botón izquierdo del *mouse*.

3.2.2.1 PairWise Charts:

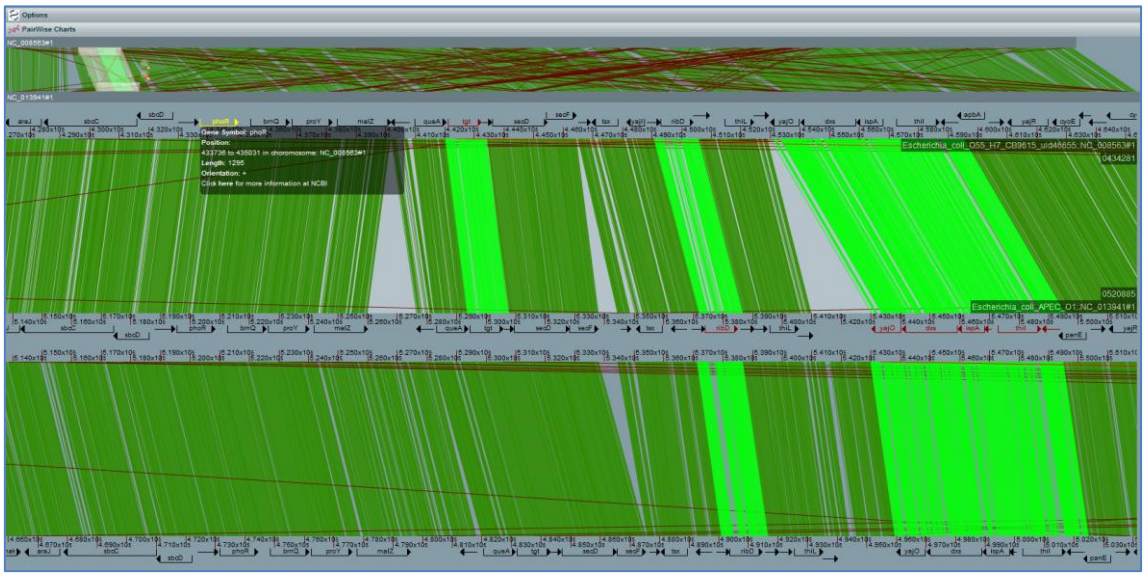

Figura 12: PairWise Charts
Parte superior: miniatura reflejando contexto de exploración del PairWise activo. Parte inferior: Concatenación de múltiples PairWise, representando las relaciones y los genes de cada genoma. Comparación entre las Bacterias Escherichia_coli_APEC_O1, Escherichia_coli_O55_H7_CB9615_uid46655 y Escherichia_coli_UMN026

El espacio de exploración de las gráficas (PairWises) se divide en dos subespacios:

- Miniatura: Espacio de interacción principal donde se visualiza el genoma (o cromosoma) en su totalidad asociado al PairWise activo.
- Grupo de PairWises ordenados según la selección del usuario. Si estas gráficas exceden el tamaño de ventana de la aplicación disponible aparece un control de scroll que permite desplazarse entre ellas. Cada uno de ellos representa la ventana (fragmento del genoma) que se está explorando. Uno de estos PairWise siempre será el elemento activo que representa la miniatura y se identifica por estar resaltado respecto a las demás gráficas.

Miniatura:

La miniatura ,siempre visible, representa el PairWise activo seleccionado en su totalidad (el genoma o cromosoma completo), ofreciendo al usuario información del contexto de trabajo. En él se hallan los principales controles de exploración (también accesibles directamente desde teclado).

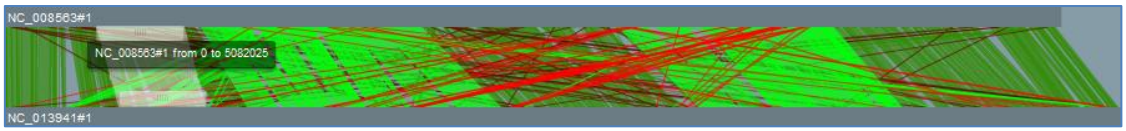

Figura 13: Miniatura, control de exploración.
El control de exploración superpuesto en la miniatura refleja el área de exploración visualizado en el PairWise activo.

En su parte superior e inferior se muestran, en caso de estar activo, las distintas regiones que corresponden cada cromosoma que compone al genoma completo que se está representando.

Las herramientas de control de exploración representadas en la miniatura permiten desplazarse por el PairWise activo, permitiendo:

- *Scroll* horizontal
- *Scroll* horizontal del genoma superior
- *Scroll* horizontal de genoma inferior
- *Zoom*

Área de exploración

El Paralelogramo blanco semitransparente representado en la miniatura corresponde al área del PairWise que se está explorando del PairWise activo.

El usuario puede desplazar el paralelogramo horizontalmente, desplazando en correspondencia el contexto de trabajado de todos los PairWises existentes (la motivación de dicho comportamiento se basa en que el usuario una vez ha determinado las diferentes regiones de interés de cada PairWise le interesa mantener las alineaciones entre estos cuando se desplaza la ventana de trabajo).

En las partes superior e inferior del paralelogramo hay dos barras de *scroll* que permiten el desplazamiento específico del genoma superior e inferior respectivamente. Esto nos permite alinear los PairWise según interese.

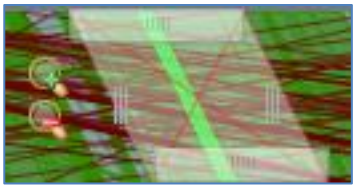

Figura 14: Detalle de la Miniatura.
Las dos líneas cruzadas reflejan la ruptura de la relación de área representada, mostrando controles alternativos.

Zoom

En sus laterales hay dos divisores que permiten al usuario arrastrar y redimensionar separadamente, por cada uno de los extremos, el tamaño de la ventana, provocando un efecto de *Zoom* (ampliación o reducción de la ventana de trabajo).

Para evitar que el tamaño del paralelogramo se hiciese demasiado pequeño, dificultando la interacción del usuario, se impuso un límite mínimo desde el cual la relación del área representada en el pairWise y la del paralelogramo se rompe.
Reflejando esta ruptura, el paralelogramo representa con dos líneas rojas cruzadas dicho estado y se muestran dos controles alternativos para operar con el *zoom* de la ventana: ampliar *zoom*(+) y reducir *zoom*(-).

El límite máximo de *zoom* se alcanza al llegar a la representación de un pixel por base nitrogenada de la secuencia del genoma.

Para mantener la coherencia de exploración los cambios de Zoom afectan a todos los PairWises existentes, y el tamaño de cada uno se expande hasta igualar el tamaño del PairWise de mayor longitud.

PairWise

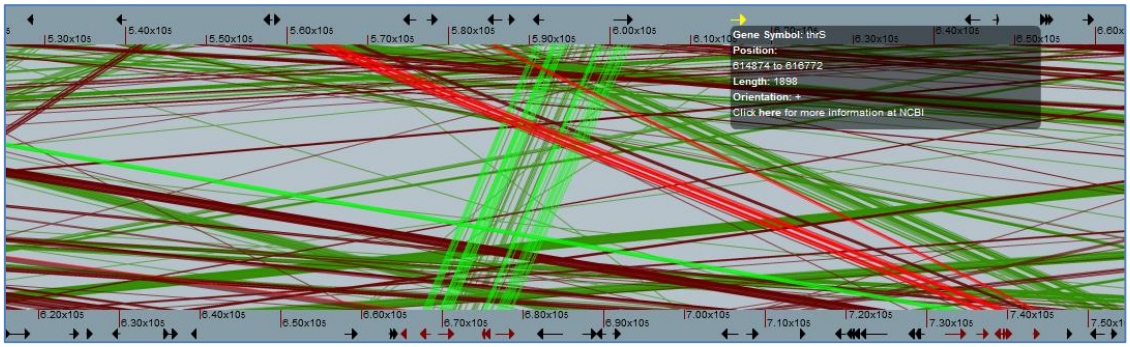

Figura 15: Detalle de un PairWise.
En verde se representan las relaciones directas, en rojo las inversas. Los fragmentos seleccionados resaltan la intensidad de su color base. Los genes seleccionados se marcan en rojo y en amarillo el que tiene el foco de información ampliada.
Comparación entre las Bacterias Wolbachia_endosymbiont_of_Brugia_malayi_TRS y Wolbachia_endosymbiont_of_Drosophila_melanogaste

Los PairWises son las representaciones gráficas entre los MUMs y SuperMUMs encontrados dados un par de genomas.
Los MUMs y SuperMUMs directos se representan en color verde.
Los MUMs y SuperMUMs inversos se representan en color rojo.
Los genes indican si es posible su símbolo identificador y la orientación de la flecha refleja el sentido de su transcripción.

Los elementos representados, en orden de arriba a abajo, son:

- Genes del genoma superior.
- Eje de coordenadas del genoma superior.
- Representación de MUMs o SuperMUMs encontrados entre el genoma superior y el inferior.
- Eje de coordenadas de genoma inferior.
- Genes del genoma inferior.

Al ubicarse el *mouse* en un PairWise se recuerda al usuario los nombres de los genomas (y cromosomas si se opera en modo cromosoma) en la zona lateral izquierda.
Se indica también la posición del puntero del *mouse* en coordenadas de las bases de cada genoma del PairWise.

Genes

El área que representa los genes, después de la miniatura, es el principal elemento de interacción del usuario. En él se opera con la selección, des-elección de genes y se accede a información ampliada de cada gen.

Ubicando el *mouse* sobre un gen este se resalta en color amarillo, y se muestra una ventana semitransparente con información detalla del gen asociado. Desde esta ventana emergente se puede acceder a la información completa de gen que ofrece el sitio web del NCBI, pulsando el enlace que se señala en la ventana se abre una nueva página hacia los detalles ofrecidos por el NCBI en el navegador.

Pulsando sobre un gen con el botón izquierdo del *mouse* se selecciona un gen, representando en rojo el estado de gen activo. La activación de un gen a su vez resalta todos los MUMs o SuperMUMs que tengan, en el genoma del gen con el que se interactúa, coincidencia en su rango de coordenadas.

El resaltar los MUMs y SuperMUMs asociados a los genes de interés del usuario ofrece una ayuda visual para encontrar las correspondencias entre ambos genomas.

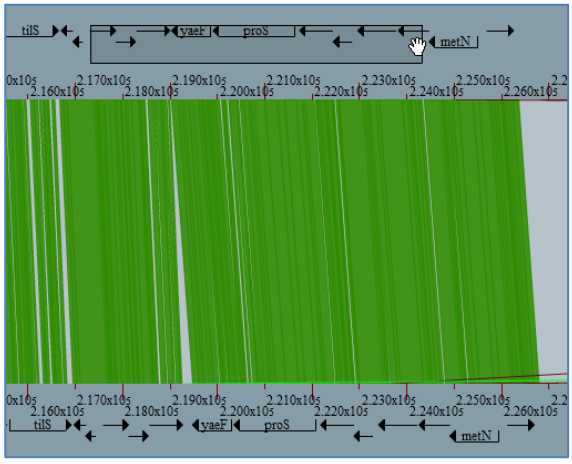

Figura 16: Multiselección de genes. Mediante el mecanismo drag-and-drop el usuario puede seleccionar en un solo paso múltiples genes.

El usuario, manteniendo pulsado el botón izquierdo del *mouse* y arrastrando hasta el punto que le interese, permite activar en un solo paso múltiples genes contiguos.

Pulsando dos veces seguidas el botón izquierdo del *mouse* en el área donde se representan los genes se des-seleccionan todos los genes previamente seleccionados.

Menú de Opciones

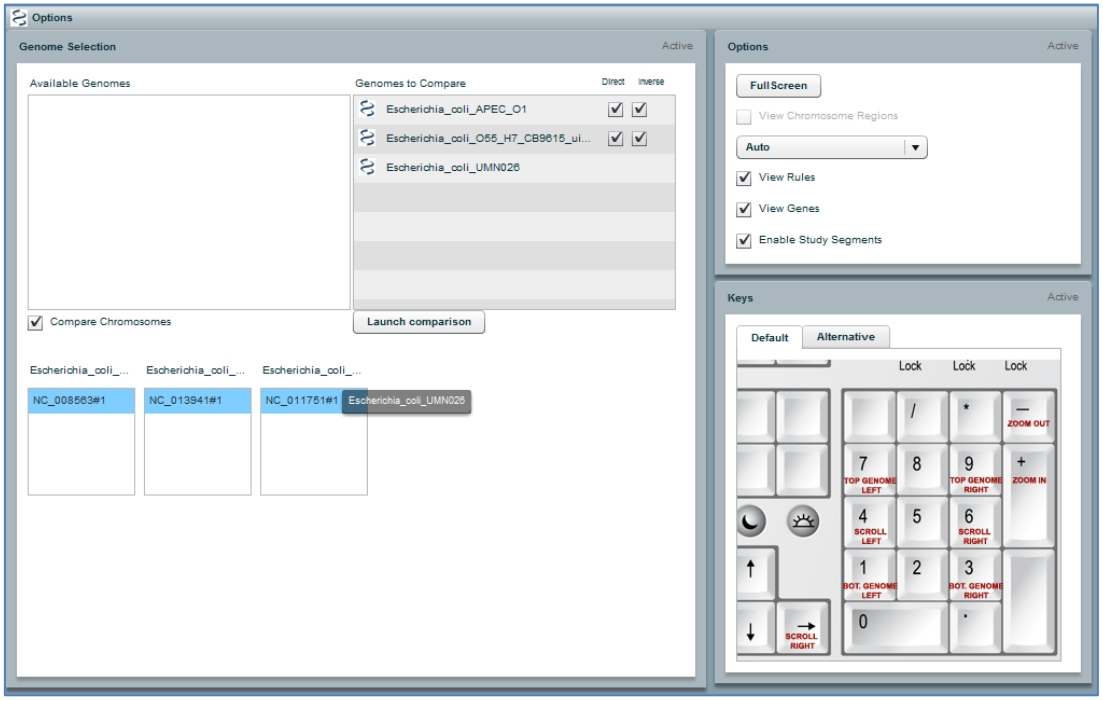

Figura 17: Selección de Genomas, Opciones de visualización y leyendas del teclado.

En el espacio de Opciones distinguimos tres grandes bloques: *Genome selection* , *Options* y *Keys.*

- *Genome selection*: El usuario selecciona el conjunto de relaciones de genomas que quiere visualizar.
    - *Available Genomes*: Listado de genomas recibidos paramétricamente de la aplicación "Mummy-tree".
    - *Genomes to Compare*: Arrastrando los genomas de interés desde "available genomes" hasta este contenedor, el usuario determina que genomas quiere comparar, el orden de estos y las relaciones a mostrar (Directas o Inversas, representando MUMs directos o inversos respectivamente).
    - *Compare Chromosomes*: Activando esta opción aparece el listado de cromosomas que componen cada uno de los genomas seleccionados. Seleccionado los cromosomas de interés, los PairWises generados representaran las relaciones entre cromosomas en lugar del genoma completo.
- *Options*: Opciones de visualización.
    - *Full Screen*: Se activa el modo de Pantalla completa de la aplicación.
    - *View Chromosome Regions*: Activando esta opción se visualiza en la miniatura (que describiremos más adelante) las distintas regiones del genoma que corresponden a cada cromosoma que lo componen.
    - *Selector*: Selección de datos a visualizar.
        - *Display Mums*: Se visualizan los MUMs.
        - *Display SuperMums+Mums*: Visualización de SuperMUMs y los MUMs detectados que no han sido absorbidos por los primeros.
        - *Display SuperMums*: Visualización exclusiva de SuperMUMs.

        - *Display Big SuperMums*: Visualización de SuperMUMs priorizando los de mayor tamaño (más adelante se detallará su significado)
        - *Auto*: La selección de los datos a visualizar se determina automáticamente en función del contexto de ventana (espacio de trabajo durante la exploración de los PairWises). Opción por defecto.
    - *View Rules*: Visualizamos o no el eje horizontal de los PairWise que indican las posiciones en bases nitrogenadas de las secuencias de los genomas.
    - *View Genes*: Visualización de los genes de cada genoma que se representa en un PairWise.
    - *Enable Study Segments*: Esta opción activa la selección de MUMs y SuperMUMs que estén comprendidos en un segmento del genoma. Estos segmentos de interés se reciben paramétricamente durante la invocación de la aplicación.
- *Keys*: Leyenda que nos indica las funciones del teclado para la exploraciones de las gráficas. Existen dos juegos de combinaciones de teclado, *Default* y *Alternative*, mientras el primero se considera el más natural e intuitivo, el segundo se ofrece como alternativo al activarse el modo a pantalla completa. Flash Player limita el acceso a los eventos de teclado en modo pantalla completa por motivaciones de seguridad.

3.2.2.2 Modo Auto: Ajuste de la visualización al rendimiento de la máquina cliente

El modo Auto de la opciones del interfaz busca ofrecer al usuario la máxima fluidez posible durante su sesión de trabajo sin perder funcionalidad, seleccionando automáticamente la mejor fuente de datos a utilizar (MUMs o SuperMUMs) y activando o no la visualización de genes.

El uso natural de la aplicación presupone el siguiente flujo de una sesión de trabajo:

- Partiendo de una visión global de los genomas se va reduciendo la ventana de exploración detectando y acotando la región de interés en cada PairWise.
- En caso de múltiples PairWise a medida que se va acotando se va alineando cada uno en función de las relaciones que tengan los MUMs o SuperMUMs.
- Llegado a cierto nivel de *zoom*, se seleccionan los genes a estudiar y se analiza sus relaciones con el resto de genomas.

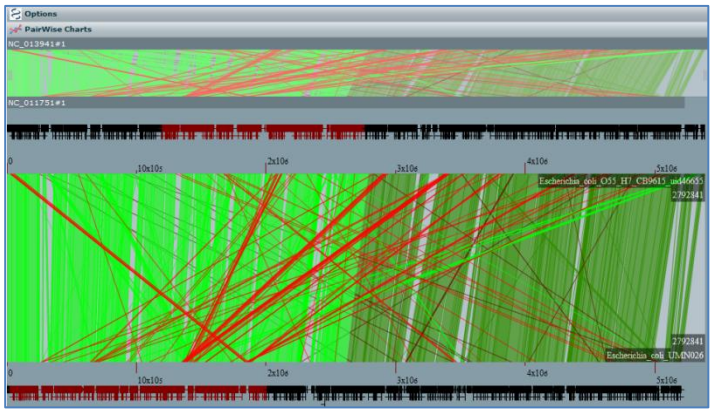

Figura 18: Representación de genes a bajos niveles de *zoom* no ofrece mucha información útil.
La alta densidad de la información representada dificulta su interacción.

Aunque explicado someramente, la lista anterior nos permite detectar los siguientes puntos que nos ayudaran a maximizar la fluidez de la aplicación:

- A bajos niveles *zoom* (tamaño de la ventana grande respecto al genoma completo) representar los MUMs no nos ofrece mayor información que la que nos pueda ofrecer la representación de SMUMs. Ya que el *skeleton* del PairWise que ambos representan a altos niveles de *zoom* es equivalente.
- Representar los genes cuando la ventana es muy grande no es útil para el usuario, ya que su representación a escala queda muy reducida y suelen tener alta densidad, complicado su interacción.
- A medida que la ventana de exploración se va reduciendo interesa ampliar el detalle de la información. Esto es, cambiar los datos representados de SuperMUMs a MUMs y visualizar los genes de cada genoma.

Por tanto, a fin de maximizar la fluidez de la aplicación el modo Auto opera de la siguiente manera:

- Durante el inicio de la aplicación se detecta la capacidad de cómputo de la máquina cliente (más adelante se detallará dicho cómputo). De dicho cómputo se extrae el número máximo aconsejable de elementos gráficos a representar, reservando un 80% de la capacidad de cómputo para procesar los genes y el 20% restante para los MUMs o SuperMUMs.
- Toda exploración inicial parte de la ventana a su tamaño máximo, representando la totalidad del genoma (o cromosoma).
- La selección de datos sigue el siguiente algoritmo:
    - Si el número de MUMs a visualizar dada la ventana actual es menor que máximo aconsejable se utilizan estos como datos a representar.
    - De otro modo y simétricamente se realiza la misma estimación con SuperMUMs y MUMs no absorbidos como fuente de datos.
    - Si no se ha seleccionado ninguna de la fuentes de datos previas, se representa tantos SuperMUMs como sean posibles priorizando los de mayor tamaño, y por tanto más representativos.
- De modo parecido la visualización o no de los genes opera del siguiente modo:
    - Dado el tamaño de ventana actual se estima el número de genes que se representarían, si este número es menor que el máximo estimado y reservado para ello se activa su visualización, de otro modo no se representan.

Para que este comportamiento tenga coherencia, hemos de resaltar que el número de SuperMUMs a representar siempre será menor que el número de MUMs que el primero agrupa.

Estos criterios permiten a la aplicación adaptarse automáticamente al contexto de trabajo, repartiendo la 'capacidad' de cómputo entre el número de PairWise a representar y considerando el volumen de datos que cada uno tenga.

De esta manera el modo Auto seguirá el siguiente comportamiento:

- Normalmente seleccionará como fuente de datos inicial SuperMUMs priorizados por tamaño (BigSMUMs), y no visualizará los genes.
- A medida que la ventana de exploración se haga más pequeña cambiará la fuente de datos representado SuperMUMs y MUMs no absorbidos.

- Cuando se llegue a niveles altos de *zoom*, finalmente la fuente de datos cambiará para representar MUMs.
- Los genes se visualizarán tan pronto como su número a procesar, determinado por el tamaño de ventana, sea menor que prefijado como aconsejable.

En caso de activar la opción por parámetro de la aplicación indicando que se está trabajando con genomas de gran tamaño, la carga de MUMs se deshabilita a fin de preservar la memoria. Y el modo Auto, única opción de exploración permitida para estos genomas, únicamente conmutará entre BigSMUMs y SuperMUMs con MUMs no absorbidos, mientras la lógica que gestiona la visualización o no de los genes permanece igual.

Cómputo

Durante el inicio de la aplicación se estima la capacidad de cómputo de la máquina cliente, para ello se generan múltiples gráficos se manera similar a los usados en la interfaz. El intervalo de tiempo consumido en este proceso junto a unos parámetros de referencia (máquina de referencia) nos permite obtener el número máximo aconsejable de elementos gráficos de tipo (S)MUMs a mostrar.

Este valor lo denotamos `_MAX_MUMS_TO_DRAW_BASE`.

`_MAX_MUMS_TO_DRAW_BASE` representando la capacidad de cómputo se reparte del siguiente modo:

```
_GENSxMUM_COSTRATIO = 1/1.3
_GEN_REQ = 0.80
_MAX_MUMS_TO_DRAW = _MAX_MUMS_TO_DRAW_BASE / number_displays


effective_max_gens_to_draw = (_GEN_REQ*_MAX_MUMS_TO_DRAW_BASE) *
_GENSxMUM_COSTRATIO
```

Si los genes no están visibles:

```
effective_max_mums_to_draw = (1-_GEN_REQ)*_MAX_MUMS_TO_DRAW
```

de otro modo:

```
effective_max_mums_to_draw = _MAX_MUMS_TO_DRAW - _malusShowingGens
```

Siendo *`malusShowingGens`* es coste 'consumido' por mostrar n genes, repartido equitativamente entre los diferentes PairWise.

`effective_max_gens_to_draw` representa el máximo número de genes a mostrar en su totalidad, todos los PairWise. Determinará la visibilidad o no de los genes.

`effective_max_mums_to_draw` representa el máximo número de (S)MUMs a mostrar para cada PairWise.

3.2.3 Estrategias de dibujado y optimizaciones

3.2.3.1 Regeneración de las gráficas

Las representaciones gráficas del Flash principalmente operan con gráficos vectoriales.

Usualmente el programador genera las gráficas vectoriales y el mismo Flash se encarga de los procesos de transformación de escalado, control de tamaños y transformación final en un *bitmap* (*array* bidimensional que contiene la información de los pixeles a representar).

En la versión previa del interfaz la estrategia de implementación se basaba principalmente en generar inicialmente los gráficos vectoriales completos de los PairWises y durante la exploración de éstas delimitar el área visualizar.

Ello provocaba que independientemente del tamaño del área explorada se generase por completo los procesos de transformación y dibujado para el PairWise, en su totalidad, para luego recortar su región visible, computando información no útil.

A medida que se ampliaba el *zoom* los tiempos de respuesta de la interfaz eran más pobres hasta llegar a los límites aceptables de usabilidad.

Ahora los PairsWises son regenerados en cada cambio del área de trabajo, provocando que sus tiempos de respuesta dependan linealmente del volumen de datos a representar en el área de exploración, computando únicamente la información que será visualizada además de permitir mayor flexibilidad para la representación de los datos visualizados.

3.2.3.2 Algoritmo base de dibujado de (S)MUMs

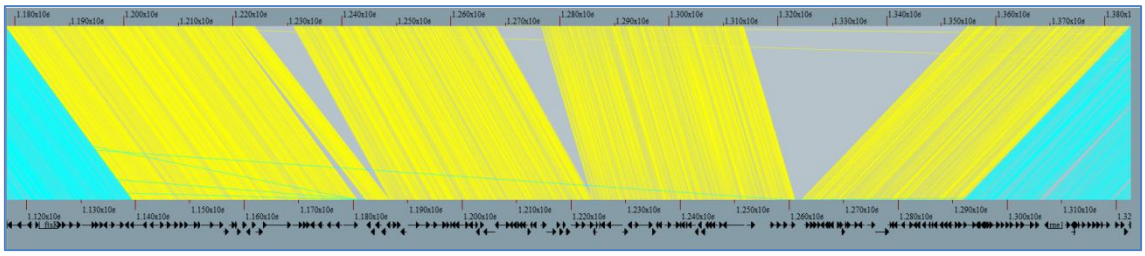

Figura 19: Visualización de las dos fases de dibujado.
En Amarillo se representa la primera fase , representando los (S)MUMs con alguna terminación en el genoma superior visible según el área de exploración. En Azul los del genoma inferior que no tengan terminación con el superior.

El algoritmo de dibujado de (S)MUMs opera del siguiente modo:

- Dibuja todos los (S)MUMs que estén en el rango superior de la ventana de exploración, asociado al genoma superior.
- Dibuja todos los (S)MUMs que estén en el rango inferior de la ventana de exploración, asociado al genoma inferior y que no tengan correspondencia al rango superior de la ventana (evitando duplicar representaciones con el primer paso)

En la Figura 19 se ha representado en amarillo los (S)MUMs que 'salen' del genoma superior , y en azul los que 'salen' del genoma inferior.

De esta manera evitamos representar los (S)MUMs diagonales que no tienen terminaciones en el rango de la ventana de exploración, ya que estas no ofrecen información de interés y dificultarían el estudio de la región de estudio.

3.2.3.3 Optimización del Algoritmo base de dibujado de (S)MUMs

Inicialmente en el algoritmo de dibujado se procesaba toda la colección de datos de (S)MUMs detectando si estaban dentro o no de la ventana de exploración para representarlos.

A medida que el volumen de datos se fue ampliando, con Millones de MUMs para Eucariotas, se mostró ineficaz, ya que el coste de procesar el vector de datos, sin considerar el dibujado en sí, ya comportaba un coste de cómputo elevado.

A coste de mayores requerimientos de memoria se optó por optimizar el proceso de filtro de datos antes de tratar para su representación.

Analizando el algoritmo de dibujado observamos que teniendo ordenados los (S)MUMs por genoma superior y por genoma inferior podemos filtrar eficientemente los datos a procesar.

Búsqueda binaria

El algoritmo de búsqueda binaria, también conocido como búsqueda dicotómica es un algoritmo eficiente de búsqueda que pertenece a la familia de algoritmos "divide y vencerás".
Sobre un *array* ordenado, subdividiendo el problema, obtiene una complejidad temporal O(log n)

Por ejemplo, en uno conteniendo 50.000.000 elementos, realiza como máximo 26 comparaciones (en el peor de los casos) hasta encontrar el elemento buscado.

De este modo, ordenando los (S)MUMS por genoma superior e inferior y usando la búsqueda binaria para encontrar el intervalo del vector de datos que corresponden a la región de la ventana de exploración, filtramos eficientemente el número de elementos a procesar para su representación visual.

El disponer de los vectores ordenados y mediante la búsqueda binaria también nos permite, para el modo Auto, estimar rápidamente el número de elementos que se visualizarían dados un vector y una ventana de exploración.

3.2.3.4 Técnicas de mejora de rendimiento en el dibujado de (S)MUMs

Los procesos de dibujado y transformación (de vector a *bitmap*) han sido detectados como los procesos más costoso de la visualización.

Para mejorar dichos procesos se han implementado las siguientes estrategias:

- Resolución interna: Discretizar el grafico vectorial
- Smooth: Desactivar efectos de suavizados durante 'transiciones'
- Proyección: Detectar y agrupar elementos gráficos que acabarán proyectados sobre los mismos puntos de la pantalla.
- Guardar en 'cache' los elementos más persistentes (sin cambios) para evitar su regeneración innecesaria.
- Uso de SuperMUMs.

Resolución interna

Al trabajar con gráficos vectoriales tenemos una capacidad de resolución casi ilimitada, pero ya que éstos se acaban representando (proyección) en un monitor (*display*) con una resolución máxima determinada, gran parte de la información se perderá, es decir, no será apreciable para el usuario.

A fin de evitar un cómputo que no aporte información se simulará una discretización de los gráficos vectoriales a unas resoluciones (resolución interna) predeterminadas.

Buscando la potencia de dos (en vistas de optimizar cálculos) más cercana a las resoluciones de monitor *'standard'*, se determinó como 1024 la amplitud y 'resolución' del grafico vectorial que representará los PairWises.

Recalcar que el grafico vectorial finalmente se escalará y representara en un *bitmap* determinado por el tamaño efectivo de ventana, es decir, la resolución de las líneas y rectángulos que representaran los (S)MUMs tendrán tanta resolución y tamaño como la pantalla del usuario ofrezca. Pero la información que en ella se representa está delimitada en 1024 'segmentos'.

Smooth

El Flash Player tiene un parámetro que permite ajustar la calidad de los gráficos representados (se puede acceder pulsando el botón derecho del *mouse* sobre una película Flash). Principalmente se encarga de activar o no el post-proceso de suavizado (*smooth*) de los *bitmaps* resultantes.

Normalmente este post-proceso se aplica como último paso de una actualización de los elementos gráficos representados.

Podemos tener un control sobre este proceso generando los *bitmaps* resultantes de los gráficos vectoriales en lugar de delegarlo al intérprete Flash.

De esta manera podemos, según interese, discriminar que elementos y cuando queremos representar en 'alta calidad' (*smooth*) o 'baja calidad' (sin *smooth*).

Llamaremos transición al estado de la aplicación que se inicia cuando el usuario interactúa con un control de exploración hasta que finaliza. Por ejemplo, mientras mantiene pulsada la tecla '+' ampliando el *zoom*.

Durante una transición podemos permitirnos representar los PairWises en 'baja calidad', ya que la vida útil de estos gráficos es de fracciones de segundo pero mantienen la información útil a representar guiando al usuario hasta la región que le interese examinar en detalle.

Para obtener mayor rendimiento durante las transiciones, puntos críticos en cómputo, la resolución interna se divide por dos, discretizando efectivamente a una recta de 512 puntos. En la Figura 20 podemos apreciar su efecto.

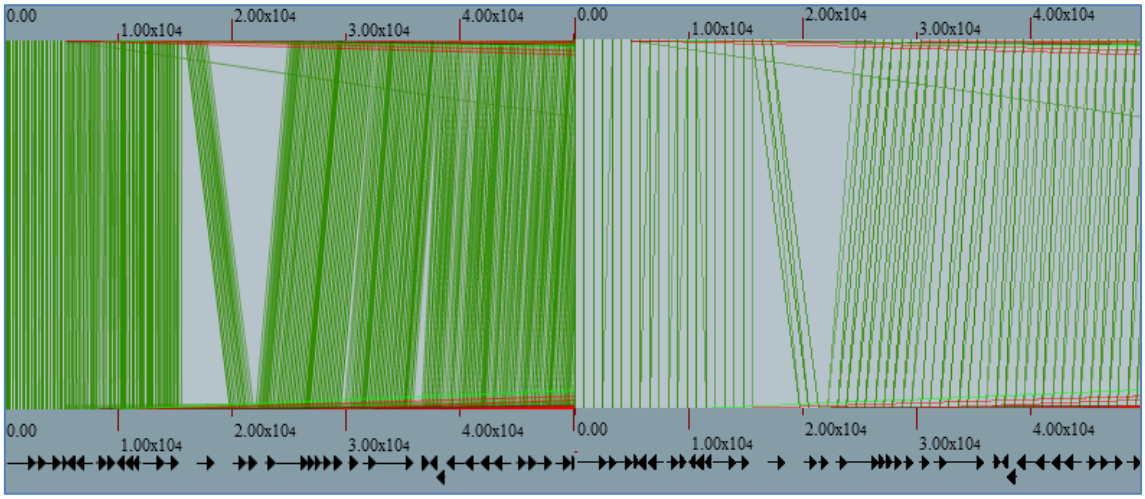

Figura 20: Impacto de resolución interna y suavizado (*smooth*).
La reducción de la resolución interna y desactivación del post-proceso de suavizado durante la manipulación de los controles de exploración nos permiten mejorar los tiempos de respuesta.

Proyección

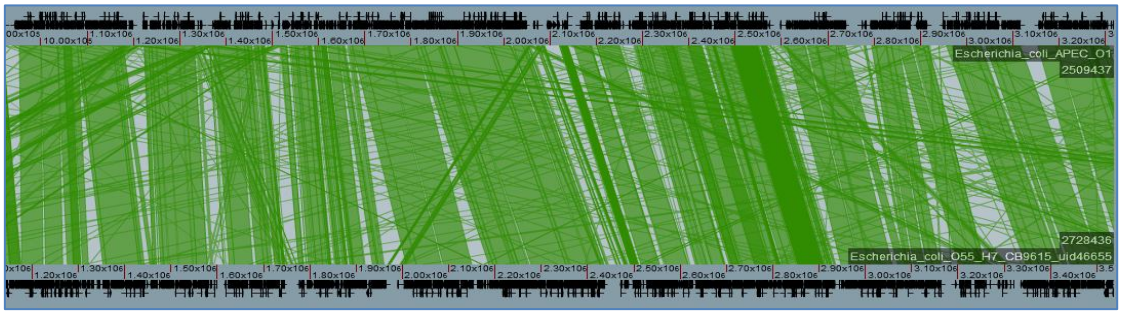

Figura 21: Información codificada por transparencia relativa.
Mediante la transparencia relativa de cada (S)MUM representado se indica al usuario donde se haya una mayor concentración de (S)MUMs.
Comparación de Bacterias: Escherichia_coli_APEC_O1 y Escherichia_coli_O55_H7_CB9615_uid46655

Para compensar la pérdida de información que provoca la discretización, que provoca que múltiples MUMs se proyecten sobre los mismos puntos, se aplica los siguientes criterios:

- En caso de superposición se representa el (S)MUM de mayor longitud.
- Se codifica mediante la transparencia del color en que se representa los (S)MUMs el número de elementos que se han proyectado en el mismo punto.
    - En cada *'render'* de detecta el número máximo de proyecciones coincidentes.

- En función de este número y la cantidad de (S)MUMs de cada elemento representado se agrupa linealmente en uno de ocho grupos de 'transparencia'.

De este modo el usuario obtiene información visual del las regiones donde hay mayor concentración de (S)MUMs.

Observar que este efecto tiene relevancia a grandes tamaños de ventana, ya que a altos niveles de *zoom* la relación número de bases representadas por pixel va tendiendo a 1.

Caché

Cuando se invalida el estado de la interfaz, el Flash vuelve a calcular todos los elementos gráficos representados.

En nuestro caso, el PairWise que representa la miniatura solo se actualiza cuando:

- Se redimensiona la pantalla.
- Se activa/desactiva (S)MUMs.
- Se cambia de PairWise activo.

A fin de evitar que se regenere innecesariamente el AS3 nos permite guardar en cache en *bitmap* previo generado.

Uso de SuperMUMs.

Complementariamente a las técnicas previas mencionadas, el uso de SuperMUMs como fuente de datos reduce en gran medida el volumen de datos a representar, manteniendo a bajos niveles de *zoom* (ventana de gran tamaño) la misma información útil (*skeleton*) que lo haría el uso de MUMs.

3.2.3.5 Representación de genes interactivos

Como hemos visto en la Tabla 5, una de las limitaciones del Flash es la capacidad de representar como máximo 16.000 instancias de símbolos, que podemos entender como representaciones gráficas de objetos.

A modo de referencia, la comparación de 9 genomas de la familia de bacterias E.Coli representamos el siguiente volumen de datos:

- 11.515 SuperMUMs y MUMs no absorbidos.
- 22.305 genes
- 323.917 MUMs

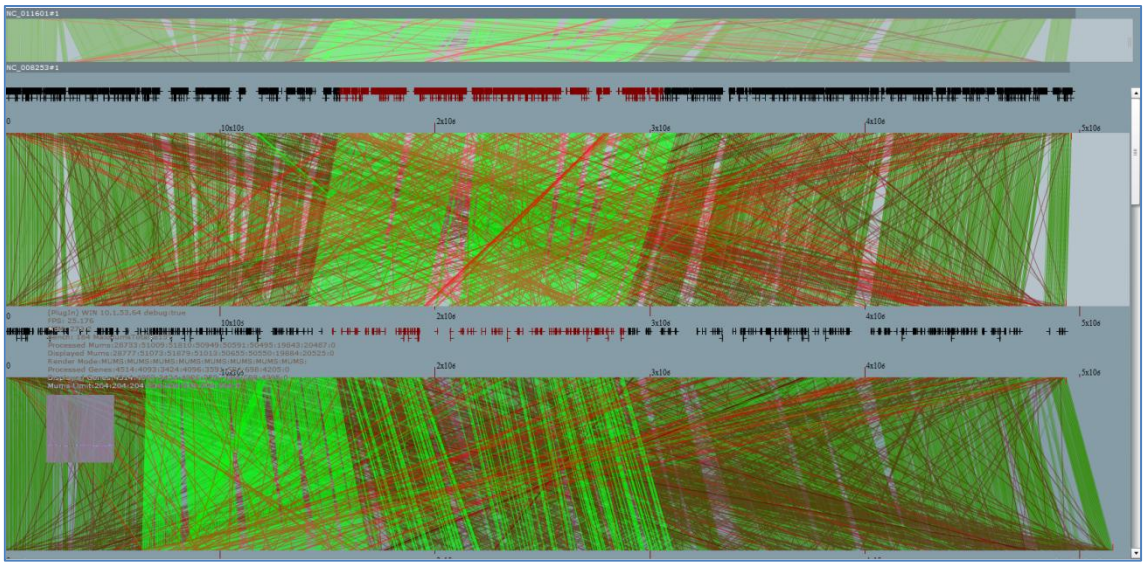

Figura 22: Comparación de 9 Bacterias mostrando genes y MUMs (la caja gris flotante es un elemento de monitorización utilizado durante el desarrollo).
El número de elementos representados e interactivos sobrepasaría las capacidades del Flash Player.
9 Bacterias: Escherichia_coli_0127_H6_E2348_69, Escherichia_coli_536, Escherichia_coli_55989, Escherichia_coli_APEC_O1, Escherichia_coli_BL21_DE3, Escherichia_coli_B_REL606, Escherichia_coli_BW2952, Escherichia_coli_C_ATCC_8739, Escherichia_coli_CFT073

Las librerías de representación de gráficas (*Charts*) que ofrece Adobe permite la interacción con cada uno de sus elementos, por ejemplo, cada línea de la gráfica es un elemento con el que se puede interactuar. Esto implica que es un elemento gráfico independiente (instancia de un símbolo) y que su número está limitado máximo está limitado a 16.000.

Al no necesitar interacción con los (S)MUMs representados pudimos representar tantos como quisiésemos, ya que creamos una única instancia de tipo *Shape* (gráfico vectorial) que manualmente transformamos y convertimos en *bitmap*. El *bitmap* resultante es un único símbolo, y delimitamos su interacción como tal, es decir, el usuario no puede interactuar con la representación de un (S)MUM concreto.

El problema surgió al implementar los genes, ya que se necesitaba la funcionalidad de interacción con cada uno de ellos, y el número de éstos sobrepasa fácilmente el límite de 16.000.

Como solución se ha creado un único elemento gráfico donde se 'dibujan' todos los genes (por genoma) y se simula la interacción con cada uno de ellos detectando las posiciones del cursor de *mouse* y buscando el gen que ha sido proyectado en esa área de pantalla.

En caso de múltiples superposiciones de genes se activa el gen más pequeño que se represente en la posición determinada por el *mouse*.

Usando esta estrategia evitamos las limitaciones de símbolos que impone el Flash a costa de una mayor complejidad de implementación para elementos interactivos.

3.2.4 Estructuras de datos

3.2.4.1 Rendimiento de las colecciones de datos

La principal estructura de datos utilizada es el nuevo tipo Vector.<type> (disponible desde la versión Flash Player 10) , en esta nueva colección de datos (tipada en contra de los clásicos ArrayCollection que trabajan con el tipo base Object) se documenta una mejora de hasta el 400% en sus operaciones de acceso/lectura respecto al resto de colecciones de datos.

La Clase MumCoord contiene la información de un (S)MUM con los que operamos.

Los datos con que mas trabajaremos serán Vector.<MumCoord>, es decir, colecciones ordenadas de (S)MUMs.

El número máximo de elementos que puede contener una colección de datos son 4,294,967,295 (2^32-1).

3.2.4.2 Procesos de ordenación

El estado inicial de exploración de la interfaz parte de la visualización completa de los genomas en cada PairWise generado.

Para genomas grandes, la gran cantidad de SuperMUMs a representar puede ser excesivo para asegurar una exploración inicial fluida. Si se detectan tales casos, el modo Auto activa la exploración *Big SuperMUMs* (mirar 3.2.2.2).

En este modo de exploración se representan tantos SuperMUMs como sea posible, en función de la capacidad de cómputo de la máquina cliente que se ha detectado previamente.

El criterio de selección de los SuperMUMs a mostrar se basa en la relevancia de éstos, que es la longitud de SuperMUM. Los SuperMUMs de mayor longitud son los más significativos a la hora de representar el *skeleton* de una comparación.

Para filtrar rápidamente los SuperMUMs más significativos necesitamos tener a éstos disponibles en una ordenación por longitud.

Para el resto de modos de exploración, necesitaremos los (S)MUMs ordenados por genoma origen y genoma destino, como hemos visto en la sección 3.2.3.3.

Se ha detectado que el proceso de ordenación por longitud que se realiza a los SuperMUMs durante la preparación de datos como el proceso más costoso. Mientras las ordenaciones por posición de genoma origen o destino son muchísimo menos costosas.

Esto es debido a que los datos ya vienen ordenados por genoma origen y existe alta correlación con la ordenación por genoma destino. En otras palabras, grandes bloques de MUMs correlativos en el genoma origen suelen encontrarse también correlativos en genoma destino.

Como veremos más adelante, por limitaciones de memoria en la exploración de grandes genomas (gran volumen de datos), solo se trabaja con datos SuperMUMs, y las ordenaciones por longitud de estos llegaban a ser demasiado costosas. Los métodos de ordenación que ofrece Flash Player son O(n log (n)).

Para solventar este problema finalmente se amplió el formato de fichero fuente de SuperMUMs añadiendo campos con los índices de ordenación bajo diferentes criterios ya pre-calculados.

3.2.5 Estudio y mejora del consumo de memoria

Utilizando la herramienta Profiler que integra el Flash Builder, analizaremos el consumo de memoria de la interfaz.

Class	Package (Filtered)	Cumulative Instances	Instances	Cumulative Memory	Memory
MumCoord	com.ibb.filomums.model.vo	324644 (90.16%)	324644 (90.26%)	18180064 (75.77%)	18180064 (75.85%)
Vector.<*>	__AS3__.vec	23310 (6.47%)	23310 (6.48%)	4751184 (19.8%)	4751184 (19.82%)
GenCoord	com.ibb.filomums.model.vo	11606 (3.22%)	11606 (3.23%)	1021328 (4.26%)	1021328 (4.26%)
MethodClosure	builtin.as$0	245 (0.07%)	0 (0.0%)	9800 (0.04%)	0 (0.0%)
HTTPOperation	HTTPService.as$33	17 (0.0%)	0 (0.0%)	2856 (0.01%)	0 (0.0%)
GenomaResource	com.ibb.filomums.model.helpers	18 (0.0%)	3 (0.0%)	2592 (0.01%)	600 (0.0%)
PairwiseVO	com.ibb.filomums.model.vo	11 (0.0%)	11 (0.0%)	2112 (0.01%)	2112 (0.01%)
Mummy		1 (0.0%)	1 (0.0%)	1884 (0.01%)	1884 (0.01%)
ContentRowChild	CanvasLayout.as$531	14 (0.0%)	0 (0.0%)	1568 (0.01%)	0 (0.0%)
MainScreen	com.ibb.filomums.view.components	1 (0.0%)	1 (0.0%)	1552 (0.01%)	1552 (0.01%)
SplashScreen	com.ibb.filomums.view.components	1 (0.0%)	1 (0.0%)	1500 (0.01%)	1500 (0.01%)
ChildConstraintInfo	CanvasLayout.as$531	14 (0.0%)	4 (0.0%)	1456 (0.01%)	416 (0.0%)
ContentColumnChild	CanvasLayout.as$531	14 (0.0%)	0 (0.0%)	1232 (0.01%)	0 (0.0%)
DirectHTTPMessageResponder	DirectHTTPChannel.as$150	17 (0.0%)	0 (0.0%)	1156 (0.0%)	0 (0.0%)
FlexFPSMonitor	com.ibb.filomums.view.components.debu...	1 (0.0%)	1 (0.0%)	1152 (0.0%)	1152 (0.0%)

Figura 23: Monitorización del uso de memoria. Herramienta disponibles en el Flash Builder.

En diferentes pruebas observamos que los elementos de mayor exigencia de memoria son: los MumCoord, GenCoord y los vectores de ordenación.

Cada instancia de MumCoord(información de un (S)MUM) ocupa 56 bytes de memoria.
Cada instancia de GenCoord (información de un gen) ocupa 88 bytes de memoria (mínimo).
Cada entrada de un Vector.<tipo> ocupa 4 bytes de memoria (puntero de referencia)

	instancias MumCoord	Instancias GenCoord	Memoria1 (Mbytes)	Memoria2 (Mbytes)	Memoria3 (Mbytes)
Test 1	380	1253	4.7	36.0	48.1
Test 2	324644	11606	28.7	62.4	75.5
Test 3	2875510	25305	198.3	234.9	247.3
Test 4	13128250	294156	¿?	1436. 0	1452.8

Memoria1: Uso de memoria (peak) tras la carga de datos según Flash Builder Profiler.
Memoria2: Uso de memoria tras la carga de datos según comando System.TotalMemory,

memoria del sistema en uso del proceso actual Flash. Ejecución en modo normal (no debug)
Memoria3: Uso de memoria tras la carga de datos según el Administrador de Tareas de Windows-7 (64bits). Ejecución en modo normal (no debug)

Test 1: 2 Arquea:: Aeropyrum_pernix, Archaeoglobus_fulgidus
Test2: 4 Bacterias: Escherichia_coli_APEC_O1, Escherichia_coli_O55_H7_CB9615_uid46655, Escherichia_coli_UMN026
Test 3: 9 Bacteria: Escherichia_coli_0127_H6_E2348_69, Escherichia_coli_536, Escherichia_coli_55989, Escherichia_coli_APEC_O1, Escherichia_coli_BL21_DE3, Escherichia_coli_B_REL606, Escherichia_coli_BW2952, Escherichia_coli_C_ATCC_8739, Escherichia_coli_CFT073
Test 4: Datos generados para el estudio del consumo de memoria.

No ha sido posible verificar la memoria consumida según el Flash Builder Profiler para el Test 4, donde se genera una excepción de tipo *'socket out'*, que normalmente indica que ha perdido el canal de comunicación con el Flash Player.

Las discordancias entre los distintas medidas de memoria se deben:

- Memoria1 y Memoria2 son medidas de el proceso actual en ejecución, nuestra interfaz. Memoria3 refleja el consumo total de todos los procesos de AVM2.
- Memoria1 refleja las instancias con alguna referencia dentro de la aplicación, la memoria efectiva con la que la aplicación está trabajando. Sin reflejar las librerías cargadas en memoria, buffers de display ni buffers de carga de ficheros.
- Memoria2 nos indica más fielmente la memoria que está consumiendo nuestro proceso, reflejando también la memoria que todavía el *Garbage Collector*(GC) no ha liberado.

Genomas Grandes

Se ha calculado como ejemplo de Eucariotas los MUMs entre el Homo Sapiens y Macaca Mulatta (primate), obteniendo del orden de 81M de MUMs.

Esto nos daría unas exigencias mínimas de 9 Gigas de memoria para sus MUMs en relación H.Sapiens →Macaca y Macaca →H.Sapiens.

Para poder trabajar con organismos del dominio Eucariota observamos que las exigencias mínimas superan las capacidades de memoria con las que puede operar el Flash.
En el apartado de mejoras y conclusiones de menciona una estrategia alternativa con la que se podrá operar sin limitaciones cualquier tipo de genomas.

Aunque con el dominio de Arquea y Bacterias no ha sido necesario, el criterio de la aplicación de cargar y pre-calcular todos los datos necesarios, entre todos los genomas recibidos por parámetro, es excesivo para el dominio Eucariota.

Si durante la sesión de trabajo el investigador analiza los genomas A→B→C, no es eficiente tener en memoria y pre-calculadas las relaciones B→A, C→B ni C→A.

Postergando la carga de datos después de que el usuario determine el tipo y orden de relaciones a estudiar permitirá un mejor aprovechamiento de la memoria disponible.

Garbage Collector

El mecanismo de gestión de memoria de AVM2 se basa principalmente en su *Garbage Collector* (GC, recolector de basura), encargado de liberar de memoria todos elementos no utilizados.

No se puede invocar al GC en una aplicación AS3 (únicamente con la versión de Flash Player Debug), por lo que no se puede tener un control directo sobre la memoria utilizada.

Se ha detectado que gran parte de la memoria de la aplicación se debe al proceso de carga y parseo de los datos, mientras que la memoria de los *bitmaps* generados durante la exploración son reciclados correctamente.

La dificultad estriba en que la monitorización del uso de memoria que ofrece Flash Builder no corresponde con la usada por AVM2, ya que, por ejemplo, la memoria temporal necesitada por los ficheros cargados no pertenece a memoria de nuestro proceso.

Una vez acabado el parseo, evitando referencia alguna a las instancias del fichero cargado para permitir su liberación no siempre funciona. No tiene métodos de liberación de memoria, y una reasignación a la instancia como *null* tampoco es efectiva para activar de modo determinístico su reciclaje de memoria.

Otro factor analizado es una posible fragmentación de memoria, que puede llevar a un uso ineficiente de esta. Las Colecciones de datos a los que se añade elementos dinámicamente son potenciales causantes de fragmentación. Tras varias pruebas no se ha podido verificar dicho fenómeno, aunque la alternativa de reservar un espacio máximo y luego recortar la memoria no usada no es atractiva ya que implicaría aumentar los requisitos mínimos de memoria a exigir.

Estos problemas, el malgasto de memoria durante la carga masiva de datos, son solventables evitando los procesos de carga de ficheros y de parseo. En el apartado de propuestas de mejora se detallará este punto.

Durante el desarrollo se cambió la política de carga de ficheros en paralelo (delegando su gestión al Flash Player) a una política de carga secuencial, a fin de estudiar su impacto en el uso de memoria.

Ello nos permitió detectar un *bug* del Flash Player que detallamos en el Anexo, a la vez que se apreció un mejor uso de la memoria y evitar la limitación que provocaba el *bug* a un uso máximo de 1Gbyte de RAM.

De hecho se implementó la adaptación de la interfaz como aplicación de escritorio, utilizando la tecnología AIR (*Adobe Integrated Runtime*) a fin de solventar el límite de memoria a 1Gbyte, pero finalmente fue descartado por no ofrecer ninguna mejora sobre su versión web una vez evitado el problema.

3.2.6 Exploración de grandes genomas

Los 81 Millones de MUMs obtenidos en la comparación entre Homo Sapiens y Macaca Mulatta , que requerirán más de 9 Gbytes de RAM, nos obliga a limitar a SuperMUMs las exploraciones

de grandes genomas. Y si estos llegan a ser excesivos, suprimir la información menos significativa hasta los límites que se estimen oportunos, ya sea por criterio de límites de memoria o evitar cargas de datos excesivamente tediosas. Este hecho se considera aceptable ya que los SuperMUMs más significativos son los que definen el esqueleto de las relaciones comparadas.

Referente al volumen de genes hallados en estos grandes genomas, no deberían representar ningún problema, ya que se han realizado simulaciones con valores cercanos a 300 mil genes y Millones de (S)MUMs, manteniendo la interfaz los buenos tiempos de respuesta.

Como referencia se estima que el genoma del Homo Sapiens contiene del orden de 25-35 mil genes, número muy inferior a las simulaciones realizadas.

El proceso más delicado, obviando las limitaciones de memoria, son los procesos de ordenación de los SuperMUMs durante el proceso de carga de datos, que podría ser excesivo con grandes volúmenes de datos. Pero al tener pre-calculados los índices de ordenación por múltiples criterios nos permite solventar este potencial problema, a la vez que aligeramos los procesos de carga de datos.

3.2.7 Pre-Proceso y Herramientas complementarias

Se han desarrollado un pequeño programa para la obtención de datos relativos a los genes de cada genoma y realizado pequeños ajustes en otros programas de la fase de pre-proceso para su adaptación a genomas grandes.

3.2.7.1 Mumol

Este programa de la fase de pre-proceso se encarga de la generación de MUMs entre dos genomas utilizando como fichero de entradas secuencias de los genomas en formato FASTA.

Adaptación para fragmentos no secuenciados

En determinadas Bacterias y Eucariotas todavía no está completada toda la secuenciación, codificando tales fragmentos con símbolos especiales.

Para adaptar el programa a estos casos, durante su fase de lectura se ha añadido un control de estos fragmentos no secuenciados, generando una tabla de desplazamientos (*offsets*) y filtrándolos en el tratamiento de datos de cómputo de MUMs.

De los MUMs resultantes se corrigen las posiciones con la tabla de desplazamientos previamente obtenidos.

Optimización de las necesidades de memoria

Otro ajuste realizado fue la minimizar el uso de memoria, para genomas pequeños usar una estructura de datos menor, concretamente:

unsigned int: 0 to +4,294,967,295 (2^32-1), 4 bytes de memoria.

en lugar de

long int: −9,223,372,036,854,775,808 to +9,223,372,036,854,775,807, -(2^63)~(2^63-1), 8 bytes de memoria

Trabajando en S.S.O.O de 64 bits.

Durante la carga de datos el programa ahora determina en función del tamaño del genoma si se puede o no operar con un tipo u otro. Optimizando las necesidades de memoria.

Las exigencias principales de memoria para la generación de MUMs son según las dos versiones:
longitud del tamaño del genoma * (64 bytes + 32 bytes)
o
longitud del tamaño del genoma * (56 bytes + 24 bytes)

Para el cálculo de MUMs el programa Mumol únicamente necesita cargar en memoria uno de los dos genomas, normalmente el más pequeño.

Particionado para genomas grandes

Para comparar dos genomas grandes como Homo Sapiens las necesidades de memoria son excesivas:

Numero de bases del genoma Humano: 3*10^9
Memoria RAM necesaria: aproximadamente 268 Gigas de RAM.

Para poder generar los datos de estudio con Eucariotas se escogió los genomas de Homo Sapiens y Macaca Mulatta, y se hizo una partición de su genoma mas grande, el Sapiens a nivel de cromosomas.

Lanzando el cómputo de MUMs entre cada cromosoma del Homo Sapiens y el genoma completo de Macaca Mulatta las exigencias de memoria se redujeron drásticamente:
El mayor cromosoma humano, Cromosoma 1 consta de 249.250.621 bases
Memoria consumida: aproximadamente 18 Gigas de RAM.

Posteriormente concatenando los ficheros parciales ajustando las ubicaciones de los MUMs de cada cromosoma se obtuvo los ficheros de MUMs finales.

Los MUMs resultantes no se pueden considerar exactos en tanto pueden existir dos MUMs generados entre diferentes cromosomas que en verdad son uno subsecuencia de otro, no respetando la definición de *Maximal Unique Matching*. Siendo necesario un post-proceso para eliminar tales casos.

Aun así ya se consideran los datos como representativos entre dos Eucariotas y válidos para el objeto de nuestro estudio, que es su comportamiento en la interfaz Mummy.

3.2.7.2 Smumsort

Este pequeño script amplia el formato de los ficheros de SuperMUMs añadiendo los índices de sus diferentes entradas según ordenaciones por genoma destino y longitud de (S)MUMs.

La motivación como se detalla en el apartado 3.2.1.1 es la de evitar el coste excesivo en cómputo durante preparación de los datos cargados, donde se ordena bajo distintos criterios las colecciones de datos de (S)MUMs.

Pre-calculando las distintas ordenaciones se consiguió aligerar significativamente el proceso de preparación de datos de la interfaz Mummy. Especialmente relevante para grandes volúmenes de datos.

Durante este proceso de pre-cómputo se aprovecha para recortar el volumen máximo de datos, eliminando los (S)MUMs menos significativos, esto es, los de menor longitud.
Actualmente se limita a 1 Millón de (S)MUMs para las relaciones directas y otro Millón para sus relaciones inversas.

3.2.7.3 MapGenes

Como parte de la fase de pre-proceso se implementó un pequeño programa en java encargado de generar los datos de los genes identificados de cada genoma.

Para Arquea y Bacterias su flujo principal es el siguiente:

- Descarga para cada genoma los distintos ficheros que ofrece el FTP del NCBI en formato GenBank (.gbk)
- Procesa cada fichero excluyendo datos no relevantes (plásmidos) y extrayendo la información relativa a los genes identificados, guardando estos en un formato propio (.gen)

Para Eucariotas el flujo es:

- Para cada genoma busca en el FTP del NCBI los diferentes subdirectorios de los cromosomas que lo componen, excluyendo los apartados relativos al ADN Mitocondrial y fragmentos aun no ubicados (*unknown*).
- Para cada cromosoma descarga el fichero en formato GenBank reducido (.gbs), que es equivalente al GenBank pero sin la información relativa a la secuencia.
  El fichero GenBank es una concatenación no ordenada de los diferentes fragmentos secuenciados de cada cromosoma.
- Dado un fichero pre-configurado (Accesions.txt), se obtiene para cada cromosoma un identificador para recuperar desde la web del NCBI información relativa al ensamblaje de cada cromosoma.
  - La url de descarga es:
    http://www.ncbi.nlm.nih.gov/sviewer/viewer.fcgi?tool=portal&sendto=on&log$

=seqview&db=nuccore&dopt=genbank&maxplex=1&val=<identificador Accession>

Su equivalente para exploración web seria:
http://www.ncbi.nlm.nih.gov/nuccore/<identificador Accession >

- La información recuperada indica las separaciones (gaps) e identificadores Accession de cada fragmento secuenciado, ejemplo:
  join(gap(10000),NT_077402.2:1..257719,gap(50000),NT_077912.1:1..153649,gap(50000),NT_004350.19:1..3323900,…
- Se construyen tablas de datos guardando tanto el orden como ubicación de cada fragmento secuenciado dentro de la secuencia completa del cromosoma.

- Se procesa cada fichero GenBank extrayendo para cada uno de los fragmentos que nos interesan (pueden haber secuencias alternativas o no ubicadas) los genes identificados, se corrigen sus ubicaciones y se guarda en un fichero temporal con el identificador de fragmento donde ha sido localizado (Accesion).
  Resaltar que la ordenación de los fragmentos dentro del fichero GenBank no tiene por qué coincidir con su ubicación dentro del cromosoma.
- Ordenadamente se concatenan los resultados parciales en el fichero .gen.

La dificultad principal estriba en la subdivisión del trabajo que realizan los investigadores en la secuenciación de los genomas, principalmente del dominio Eukarya.
Los fragmentos secuenciados que componen cada cromosoma están etiquetados con un identificador Accesion, y estos no contienen información relativa al contexto donde se hallan.
Los ficheros GenBank por si mismos no contiene información para el ensamblaje completo del mismo.
Existen secuenciaciones alternativas, etiquetajes temporales, fragmentos desconocidos, etc... Y las secuencias 'oficiales' (*primary reference assembly*) no tienen por qué ser completas, de ahí los múltiples gaps entre los distintos fragmentos secuenciados.

4. Informe técnico

4.1 Mummy

Aplicación desarrollada en AS3, con la herramienta de desarrollo Flash Builder 4 usando Flex SDK 4.1.

Parámetros

Parámetros recibidos en formato HTTP GET:

- V<indice_genoma> = <codigo_genoma>
- NombreVar<indice_genoma> = <nombre_del_genoma>
- path = <URL_ficheros_de_datos>
- [baseini<indice_genoma> = <base_inicial_segmento1>,<base_inicial_segmento2>,…]
- [basefin<indice_genoma> = <base_ final _segmento1>,<base_final_segmento2>,…]
- [smumsOnly = true]

Los índices del genoma, números enteros de 1 a 9, se esperan ordenados, es decir, si se quiere parametrizar tres genomas, estos se esperan llegar etiquetados como: 1,2 y 3.

El código de genoma es un número entero asociado al genoma, los ficheros de datos de dicho genoma se identifican por dicho código.

El nombre del genoma es espera con caracteres alfanuméricos (0 al 9,A a la Z). y carácter "_" como separador entre palabras.

Las bases de los segmentos han de ser números enteros, con el mismo número de elementos por cada genoma en su parámetro baseini y basefin donde la base final de cada segmento ha de ser superior a su base inicial correspondiente.

El parámetro *path* recibe la *url* de los ficheros de datos, esta puede ser absoluta o relativa.

El parámetro *smumsOnly* activa el modo para grandes genomas, cargando únicamente la información de SuperMUMs y genes, descartando los MUMs.

Ficheros de datos

La estructura de directorios indicado por el parámetro path ha de tener el siguiente formato:

directorios	ficheros	descripción
/mums		Directorio contenedor de ficheros de datos relativos a los MUMs
	<a>_<b>.directo	Listado de MUMs entre genoma <a> y <b> en relación directa

	<a>_<b>.inverso	Listado de MUMs entre genoma <a> y <b> en relación inversa
/smumsort		Directorio contenedor de ficheros de datos relativos a los SuperMUMs, formato ampliado
	<a>_<b>.directo.smum	Listado de SuperMUMs y MUMs no absorbidos entre genoma <a> y <b> en relación directa
	<a>_<b>.inverso.smum	Listado de SuperMUMs y MUMs no absorbidos entre genoma <a> y <b> en relación inversa
/mapedgenes		Directorio contenedor de ficheros de datos relativos a los genes de cada genoma.
	<a>.gen	Listado de genes del genoma <a>
	<b>.gen	Listado de genes del genoma <b>

Ficheros de MUMs:

Cada línea del los ficheros .directo e .inverso ha de tener el siguiente formato, con el carácter espacio (0x20) como separador y la secuencia 0x0D,0X0A para cambios de línea:

<base de inicio del MUM del genoma a> <base de inicio del MUM del genoma b> <longitud del MUM>

El fichero se espera ordenado , de menor a mayor, por los valores de su primera columna.

Ficheros de SuperMUMs, formato ampliado:

Cada línea del los ficheros .directo.smum e .inverso.smum ha de tener el siguiente formato, con el carácter espacio (0x20) como separador y la secuencia 0x0D,0X0A para cambios de línea:

<base de inicio del (S)MUM del genoma a> <base de inicio del (S)MUM del genoma b> <longitud del (S)MUM> <etiqueta de SuperMUM o MUM no absorbido [s|m]> <índice de ordenación por genoma b> <índice por ordenación por longitud>

El fichero se espera ordenado , de menor a mayor, por los valores de su primera columna.

Ficheros de Genes:

Cada línea del los ficheros .gen ha de tener una de siguiente tipo de entradas, con el carácter tabulador (0x09) como separador y la secuencia 0x0D,0X0A para cambios de línea:

C <Nombre/Código del Cromosoma> <longitud del Cromosoma>

G <base inicio gen> < base final gen > < orientación [+|-] > < identificador del gen según la bbdd del NCBI> <símbolo identificador del gen>

El primer tipo de entrada, precedido por el carácter 'C', identifica a un cromosoma del genoma que describe. El orden de los cromosomas ha de corresponder con el de la secuencia del genoma. Todos los genes descritos a continuación de una entrada de tipo 'C' (cromosoma) se agrupan en este (los genes pertenecen a ese cromosoma).

El fichero se espera ordenado , por cromosomas en orden creciente y genes por su base de inicio, también por orden creciente..

Ficheros del código

Basándonos en el FrameWork PureMVC, la estructura de los ficheros es la siguiente:

Directorios	Ficheros	Descripción
/		
	Mummy.mxml	Aplicación Mummy
/assets		Recursos gráficos, css,..
/com.ibb.filomums		
	ApplicationFacade.as	Fachada de la
//controler		
	ApplicationStartupCommand.as	Control secuencia incial de comandos
	BenchCommand.as	Computo de Benchmark
	ModelPrepCommand.as	Preparación de modelo de datos
	ViewPrepCommand.as	Preparación de las vistas
//model		Proxies del modelo de datos
	ConfigProxy.as	Configuración
	GenomesProxy.as	Control de Genomas
	LocaleProxy.as	Configuración de idiomas (no usado)
	MapedGenProxy.as	Genes y Cromosomas
	MumsProxy.as	Datos MUMs
	OptionsProxy.as	Opciones de visualización
	ParamsProxy.as	Parámetros de la aplicación
	SMumsProxy.as	Datos SMUMS
	StartupMonitorProxy.as	Monitor de carga de proxies
///business		
	LoadGenDelegate.as	Carga de ficheros de genes
	LoadGenomaDelegate.as	Carga de ficheros de (S)MUMs
	LoadXMLDelegate.as	Carga de ficheros XML (configuración)
///helpers		
	GenomaResource.as	Parser (S)MUMs
	GenResource.as	Parser de Genes
	XmlResource.as	Parser de ficheros configuración (XML)
///vo		Value Objects
	CromosomaVO.as	Datos Cromosoma
	DisplayEntry.as	Datos control de MUMs a visualizar
	GenCoord.as	Datos Gen
	ListItemVO.as	Datos elemento lista del menú opciones
	MumCoord.as	Datos (S)MUMs

	PairwiseVO.as	Datos de un PairWise
	ResourceVO.as	Datos de un recurso a cargar
	ScrollRangeVO.as	Datos de estado de ventana de exploración
	StudyRegionVO.as	Datos de Segmentos de estudio a estudiar
//view		
	ApplicationMediator.as	Mediador de la aplicación
	MainScreenMediator.as	Mediador de la página principal
	SplashScreenMediator.as	Mediador de la página de carga
///components		
	MumGraph.as	Render de un PairWise
	SplashScreen.mxml	Diseño de la página de carga
	MainScreen.mxml	Diseño de la página principal
	GraphScroller.mxml	Diseño y gestión del control de miniatura
	Divisors.mxml	Diseño de la miniatura
	RangeActiu.as	Render del paralelogramo de la miniatura

4.2 MapGenes

Pequeña aplicación en Java que dado un conjunto de genomas busca de en FTP de NCBI los ficheros de datos pertinentes para generar la información de genes que alimentan a la interfaz.

Parámetros

java -Xmx512m -jar getgenes.jar -type=[Archaea|Bacteria|Eukaryota] -downloadfiles=[true|false]

El parámetro *type* determina el dominio al que pertenecen los genomas a procesar.
El parámetro *downloadfiles* determina si es o no necesaria la descarga de datos para procesarlos (ya que se puede disponer previamente de ellos).

Configuración

El fichero *config.properties* ubicado en el directorio /bin contiene las siguientes entradas:

```
remote_host=ftp.ncbi.nlm.nih.gov
local_directory_Archea=/www/applic/genomas/genomas/Archaea/
local_directory_Bacteria=/www/applic/genomas/genomas/Bacteria/
local_directory_Eukaryota=/www/applic/genomas/genomas/Eukaryota/
remote_directory_Archea=/genomes/Bacteria/
remote_directory_Bacteria=/genomes/Bacteria/
remote_directory_Eukaryota=/genomes/
```

remote_host: URL de la FTP del NCBI donde recuperar los datos
local_directory_Archea: Directorio local donde se espera encontrar el fichero genes.txt con el listado de nombres de los genomas Arquea a procesar (nombre del genoma por línea)
local_directory_Bacteria: Directorio local donde se espera encontrar el fichero genes.txt con el listado de nombres de los genomas a procesar (nombre del genoma por línea)
local_directory_Eukaryota: Directorio local donde se espera encontrar el fichero genes.txt con el listado de nombres de los genomas a procesar (nombre del genoma por línea)
remote_directory_Archea: Directorio remoto del FTP donde se ubican los

genomas de tipo Arquea.
remote_directory_Bacteria: Directorio remoto del FTP donde se ubican los genomas de tipo Bacteria.
remote_directory_Eukaryota: Directorio remoto del FTP donde se ubican los genomas de tipo Eukaryota.

Por cada genoma del tipo Eukaryota a procesar ha de existir un subdirectorio de la aplicación con el siguiente formato:
./assemblies/<nombre_genoma>/
Conteniendo el fichero Accession.txt con el siguiente formato por línea:

<Identificador de Cromosoma> <RefSeq><Accession.version>

Siendo RefSeq y Accession.version distintos identificadores de la BBDD del NCBI de los ficheros GenBank asociados a cada cromosoma.

Este fichero se suele encontrar, bajo diferentes nombres, en:
ftp://ftp.ncbi.nih.gov/genomes/<nombre_genoma>/Assembled_chromosomes/

Resultado

El conjunto de ficheros resultantes, ficheros en formato .gen, se ubican en el subdirectorio: ./genomas/Archaea/mapedgenes ,./genomas/Bacteria/mapedgenes o ./genomas/Eukaryota/mapedgenes según el tipo de genomas a procesar.

Librerías

edtftpj

edtftpj es una librería ofrecida por Enterprisedt[8] que permite el control programático de sesiones ftp en Java.

BioJava

BioJava es un conjunto de librerías en java para el proceso de datos biológicos. En nuestro caso lo utilizamos para parsear los ficheros en formato GenBank [9]de la que nos interesa extraer los datos de genes detectados de un genoma.

4.3 Pre-Proceso

[8] http://www.enterprisedt.com/products/edtftpj/

[9] GenBank es un formato de fichero de información de secuencias nucleótidos y de sus transcripciones proteínicas ofrecido por el NCBI.

La generación de datos consumidos por la aplicación Mummy -tree y Mummy son computados previamente por un conjunto de scripts y programas en servidores de la familia Unix.

La secuencia principal del pre-proceso es la que sigue:

- Gestio_Genomas (./Gestio_Genomas/Gestio_Genomas): Programa de descarga y ensamblaje de las secuencias de los genomas (ficheros FASTA) desde el FTP del NCBI.
- lanzadera.sh (./lanzadera/lanzadera.sh): Script que automatiza la llamada a los siguientes scripts.
    - lanzadera_archaea.sh (./lanzadera/lanzadera_archaea.sh): Computo de pre-proceso para genomas del dominio Arquea.
    - lanzadera_bacteria.sh (./lanzadera/lanzadera_bacteria.sh): Computo de pre-proceso para genomas del dominio Bacteria.
    - lanzadera_eukaryota.sh (./lanzadera/lanzadera_ eukaryota.sh): Computo de pre-proceso para genomas del dominio Eukarya.

La secuencia de cada script lanzadera_<dominio_genomas> es:

- rename.sh (./Gestio_Genomas/rename.sh): Renombra los genomas por números identificativos.
- lanza_mums (./lanza_mums/lanza_mums): Procesamiento de las secuencias generando los ficheros de datos de MUMs (./lanza_mums/mums/) , SuperMUMs (./lanza_mums/smums/) y calculo de factores.
- smumsort.sh (./lanza_mums/smumsort.sh): Calculo de índices para SuperMUMs y recorte de tamaño si se sobrepasa cierto máximo número de entradas (elimina los SuperMUMs menos significativos). Genera la ficheros de SuperMUMs ampliados en ./lanza_mums/smumsort/
- mapgenes (./mapgenes/mapgenes): Descarga y extracción de datos desde el FTP del NCBI para obtención de los genes identificados en cada genoma.
- factors2samples (./factors2samples/factors2samples): Generación de factores ordenados.
- correl (./millor_correlacio/correl): Computo de correlaciones entre genomas.
- Prim (./mstree/Prim): Computo del Minimum Spanning Tree de las relaciones entre genomas.
- clusters (./clusters/clusters): Generación de Clusters.
- grafoclusters_genoma (./grafoclusters_genoma/ grafoclusters_genoma): Generación de grafoclusters.

Los datos resultantes se ubican según cada tipo de genomas en:

- /genomas/Archaea/
- /genomas/Bacteria/
- /genomas/Eukariota/

Estructurados del siguiente modo:

directorios	ficheros	descripción
/genome		Secuencias de los genomas procesados
	<a>	Secuencia de genoma <a> en formato FASTA
/mums		Directorio contenedor de ficheros de datos relativos a los MUMs
	<a>_<b>.directo	Listado de MUMs entre genoma <a> y <b> en relación directa
	<a>_<b>.inverso	Listado de MUMs entre genoma <a> y <b> en relación inversa
/smums		Directorio contenedor de ficheros de datos relativos a los SuperMUMs
	<a>_<b>.directo.smum	Listado de SMUMs entre genoma <a> y <b> en relación directa
	<a>_<b>.inverso.smum	Listado de SMUMs entre genoma <a> y <b> en relación inversa
/smumsort		Directorio contenedor de ficheros de datos relativos a los SuperMUMs formato ampliado
	<a>_<b>.directo.smum	Listado de SMUMs entre genoma <a> y <b> en relación directa
	<a>_<b>.inverso.smum	Listado de SMUMs entre genoma <a> y <b> en relación inversa
/mapedgenes		Directorio contenedor de ficheros de datos relativos a los genes de cada genoma.
	<a>.gen	Listado de genes del genoma <a>
	<b>.gen	Listado de genes del genoma <b>
/factors_ordenats		Factores ordenados de los genomas
	<a>	Listado de factores del genoma <a>
./		
	BioFactors.net	Resultados del programa grafoclusters_genoma
	BioMST.net	Resultados del programa Prim
	clusters.txt	Resultados del programa clusters
	factors.txt	Factores computados por lanza_mums

	genes.txt	Listado de los nombres de los genomas. La ordenación determina el indicie asignado a cada genoma.
	millor_correlacio.txt	Resultados del programa correl
	mstprim.txt	Resultados del programa Prim

5. Conclusiones

Los objetivos marcados han sido sobradamente alcanzados. El conjunto de estrategias aplicadas nos aseguran una exploración fluida para cualquier volumen de datos que hayamos conseguido cargar en memoria, pudiendo comparar genomas de arquea, bacteria y eucariotas de miles de millones de bases.

Hoy por hoy, podemos asegurar que no existe ninguna interfaz web equivalente que permita una exploración amigable y fluida. Dado que el flujo de información suministrada por Mummy se adapta a la capacidad de la máquina cliente, al tamaño de los genomas y al número de genomas que se están comparando simultáneamente, siempre se consiguen el máximo de prestaciones en cada consulta. De esta forma, Mummy facilita para cualquier máquina cliente desde una visión global del esqueleto de conservación entre los genomas, así como un seguimiento gen a gen, sin importar el tamaño de estos.

El diseño de la interfaz también ha sido elaborado con el máximo cuidado, adaptándolo a las necesidades de trabajo del investigador que ha de utilizar Mummy, con el fin de maximizar su usabilidad.

Para testear la herramienta se han generando simulaciones con 8 Millones de (S)MUMs y 294 mil genes. Los resultados han sido óptimos, en el sentido que la interfaz ha funcionado fluidamente durante su exploración. También se han computado un conjunto de comparaciones entre genomas reales representando cada dominio de la vida: Arquea, Bacterias y Eucariotas, que han acompañado las ilustraciones de esta memoria.

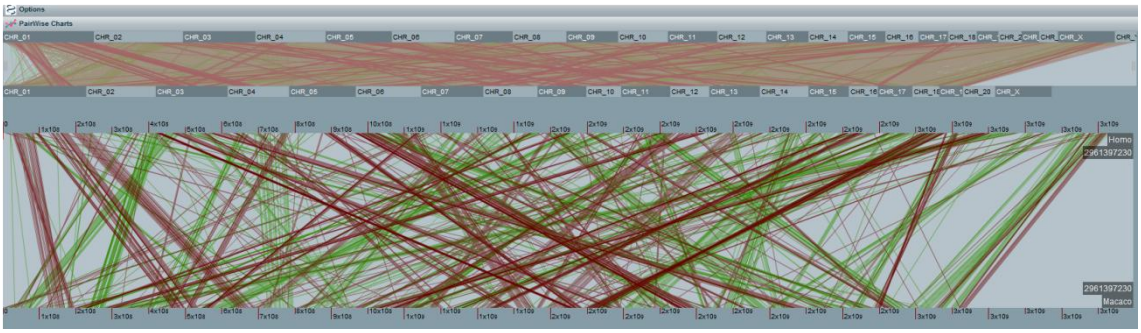

Figura 24: Comparación entre dos Eucariotas: Homo Sapiens (3000M de bases) y Macaca Mulatta (2900M de bases).
Para el estudio de genomas grandes solo está accesible la exploración de SuperMUMs, manteniendo estos la información mas relevante del esqueleto de la comparación.

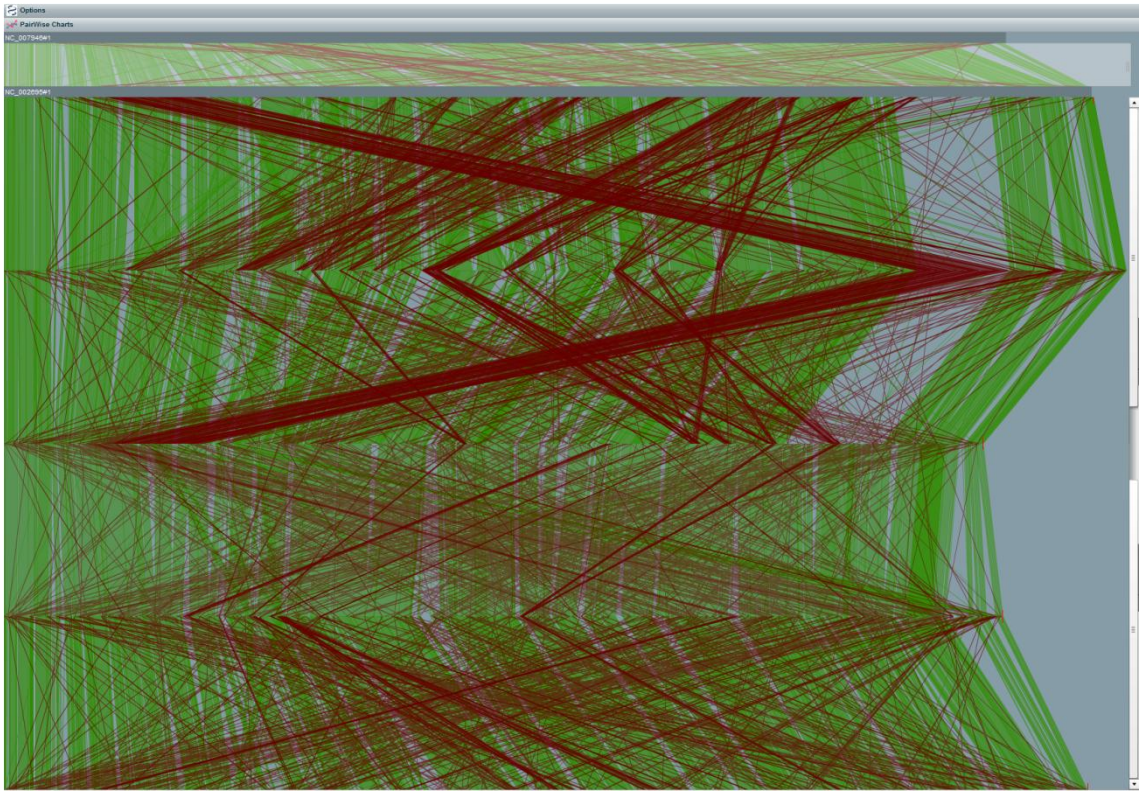

Figura 25: Comparación de 5 Bacterias de la familia E.Coli: Escherichia_coli_O157_H7_TW14359 (5.6M de bases), Escherichia_coli_O26_H11_11368 (5.7M de bases), Escherichia_coli_0127_H6_E2348_69 (5M de bases), Escherichia_coli_UTI89 (5.1M de bases), Escherichia_coli_O157H7 (5.5M Bases)

Los ambiciosos objetivos que nos habíamos marcado han sido alcanzados en su totalidad:

- Crear una interfaz web que mediante el uso de MUMs y SuperMUMs permita comparar genomas completos.
- Permitir tanto una visión global de las relaciones (esqueleto) como llegar a un detalle a nivel de bases.
- Permitir que cuando no todo el genoma sea visible se pueda decidir en qué área de la comparación se ubica el área de visualización. Permitir visualizar la comparación entre cualquier parte de un genoma con cualquier parte del otro genoma en el área de visualización para cualquier *zoom*.
- No mostrar información no útil, como MUMs diagonales cuyo punto de origen y destino no estén dentro del área de visualización activa.
- Crear un programa que recupere, procese y ensamble los genes identificados de un genoma, detectando los cromosomas en los que están ubicados.
- Mostrar en la interfaz los genes de los genomas involucrados en cada comparación, permitiendo acceder a una información ampliada de cada gen.
- Permitir la selección de genes de manera individual o múltiple resaltando la correspondencia de este gen con el otro/otros genoma (MUMs y SuperMUMs).

- Aceptar regiones de los genomas por parámetro para resaltar la conservación de ese genoma en el otro/otros genomas (MUMs y SuperMUMs).
- Adaptar la fase de pre-proceso de generación de datos para la comparación de grandes genomas y optimizar el consumo de memoria del cálculo de MUMs para estos grandes genomas.
- Conseguir representar en la interfaz cualquier tipo de genoma (Arquea, Bacteria y Eucariota).
- Permitir en la interfaz la comparación múltiple de 9 genomas simultáneamente.
- Permitir trabajar a nivel de cromosoma, acotando la exploración de un genoma completo a los cromosomas que lo componen.
- Optimizar la interfaz a fin de obtener buenos tiempos de respuesta durante la exploración de las comparaciones.
- Ajuste automático de la información a mostrar en función de la capacidad de cómputo de la máquina cliente.
- Maximizar el tamaño de las gráficas para una mayor accesibilidad a la información representada.
- Diseño líquido, adaptable al tamaño de ventana disponible.
- Permitir trabajar a pantalla completa.

Más allá de la experiencia adquirida y del aprendizaje de aspectos técnicos de la tecnología utilizada, me considero orgulloso de haber ideado un conjunto de estrategias gráficas que han resultado efectivas para solventar los distintos problemas a encarar. Problemas que no eran pocos ni pequeños.

Que el esfuerzo realizado haya producido una herramienta de características novedosas en un campo científico totalmente puntero y de rabiosa actualidad nos congratula (a mí y a mi director), y nuestro mayor deseo es que sea de utilidad para la comunidad científica a la que va dirigida.

5.1 Propuestas de Mejora

La versión actual se considera eficiente y completa para operar con Arquea y Bacterias, para operar a nivel de MUMs con Eucariotas se tendrían que cambiar algunas estrategias.

Las limitaciones actuales principalmente podemos resumirlas en:

- Límite de memoria
- Coste excesivo de carga y preparación de datos para altos volúmenes de datos.

El Flash permite la carga de datos remota mediante serialización de objetos nativos en AS3.

Una de las múltiples herramientas que simplifica este proceso es la aplicación WEBORB, actuando como proxy en el lado servidor (disponible para PHP,.NET, Java y Rails), convierte instancias de clases nativas del lenguaje del servidor en AS3 y las serializa mediante AMF3 (*Action Message Format versión 3*).

Si se procede a la carga de datos después de la selección de genomas, solo sería necesario cargar los datos de dicha sesión de trabajo, y no toda la posible combinatoria que se puede establecer dado un conjunto de genomas.

Cargando estos datos serializados (*Flash remoting objects*) conseguimos:

- Optimizamos uso de memoria al cargar únicamente los datos necesarios. Evitamos carga de ficheros para obtención de datos.
- No hay necesidad de parsear los datos obtenidos.
- Procesos costos como mapeado de relaciones (S)MUMS con genes puede venir ya pre-calculado.
- El volumen de datos transferidos es muy inferior que usando ficheros texto.

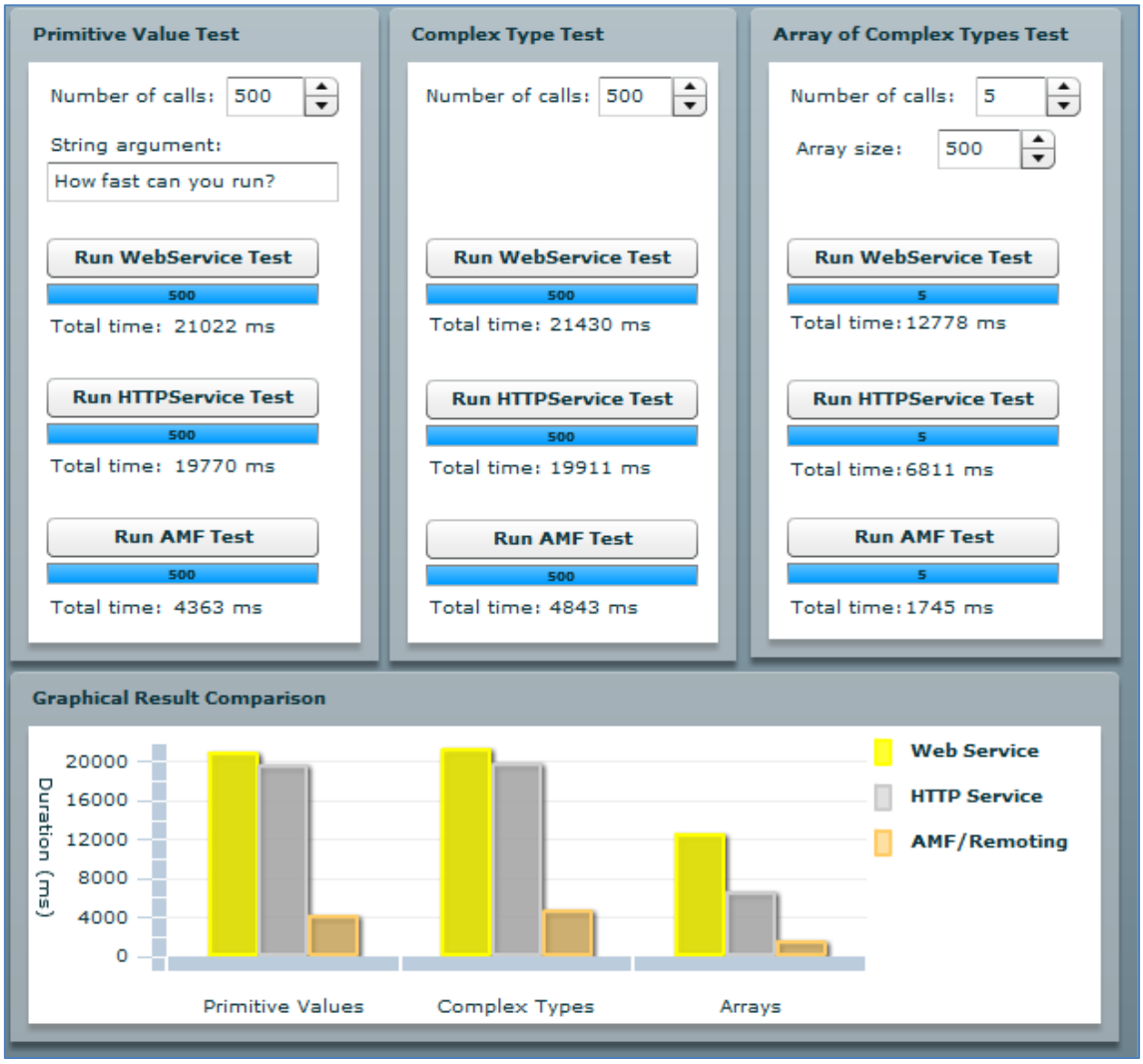

Figura 26: Comparativa rendimiento entre Web Service, Http Service y AMF (Flash Remoting)

Ya que no es viable tener cargados en memoria los genomas completos se tendría que mantener en memoria únicamente la ventana de datos necesarios que establezca la ventana de

exploración, forzando que cuando esta ventana sea demasiado grande únicamente es posible su exploración de BigSMUMs.

WEBORB y herramientas similares ofrecen estos servicios conocidos como servicios *Paging* (paginación de datos).

6. Referencias

[1] **Efficient Space and Time multicomparision of Genomes,** Mario Huerta, Xavier Messeguer, Technical report LSI-02-64-R. LLenguatges i Sistemes Informatics, Universitat Politècnica de Catalunya (2002).
[2] **Alignment of Whole Genomes**, Delcher A. .L., Kasif S., Fleischmann R. D., Peterson J, White O., Salzberg S.L., Nucleic Acids Research, 27:11, 2369-2376. (1999)
[3] **Identification of patterns in biological sequences at the ALGGEN server: PROMO and MALGEN**, Domènec Farré, Mario Huerta, Romà Roset, José E. Adsuara, Llorenç Roselló, M. Mar Albà, and Xavier Messeguer, Nucleic Acids Research. 2003 31: 3651-3653 (2003).
[4] **M-GCAT: Interactively and efficiently constructing large-scale multiple genome comparison frameworks in closely related species.** T. Treangen and X. Messeguer. BMC Bioinformatics 2006, 7:433.(2006)
[5**] Mauve: multiple alignment of conserved genomic sequence with rearrangements.** Aaron C.E. Darling, Bob Mau, Frederick R. Blatter, and Nicole T. Perna. 2004. Genome Research. 14(7):1394-1403.(2004)
[6] **CoCoNUT: an efficient system for the comparison and analysis of genomes BMC Bioinformatics** ,Mohamed I. Abouelhoda, Stefan Kurtz, Enno Ohlebusch , 9:476, (2008).

Figuras

Figura 1,6: http://www.uv.es/metode/anuario2001/184_2001.html
Figuras 2,3,4: http://html.rincondelvago.com/sintesis-de-proteinas.html
Figura 5: http://es.wikipedia.org/wiki/%C3%81rbol_filogen%C3%A9tico
Figura 10: http://puremvc.org/component/option,com_wrapper/Itemid,195/
Figura 26: http://www.themidnightcoders.com/products/weborb-for-net/developer-den/technical-articles/amf-vs-webservices.html

Tablas

Tabla 2: http://www.uv.es/metode/anuario2001/184_2001.html
Tabla 4: http://www.adobe.com/products/player_census/flashplayer/version_penetration.html
Tabla 5: http://kb2.adobe.com/cps/144/tn_14437.html

Recursos Web relevantes

Wikipedia:
http://www.wikipedia.org

Adobe:
http://livedocs.adobe.com/

Comunidades de desarrolladores:
http://www.madeinflex.com/
http://www.actionscript.org/forums
http://osflash.org/as3_speed_optimizations

Especificaciones varias:
http://biojava.org/wiki/Main_Page
http://www.ncbi.nlm.nih.gov

Anexo

Bug del Flash Player:

Durante el desarrollo se encontró con un el límite de memoria máximo a utilizar en valores cercanos a 1 Gbyte de RAM (memoria del proceso), generándose el evento "*out of memory*" (sin memoria disponible).

Para verificar que no fuese un límite impuesto por los browsers se procedió a adaptar la aplicación como AIR, y cuando la aplicación AIR se acercaba al 1 Gbyte de memoria se detectó el siguiente error:

adl.exe (AIR Debug Launcher)

Unhandled exception at 0x645e84bd in adl.exe: 0xC0000005: Access violation writing location 0x00000008.

```
645E84BD  mov       word ptr [esi],bp
645E84C0  add       esi,2
645E84C3  sub       edx,1
645E84C6  mov       dword ptr [esp+1Ch],edx
645E84CA  add       edi,1
645E84CD  add       dword ptr [esp+10h],1
645E84D2  test      edx,edx
645E84D4  jg        645E8341
645E84DA  mov       eax,dword ptr [esp+10h]
645E84DE  pop       edi
645E84DF  pop       esi
645E84E0  pop       ebp
645E84E1  pop       ebx
645E84E2  pop       ecx
645E84E3  ret
```

Durante el desarrollo se cambió la política de carga de ficheros en paralelo (delegando su gestión al Flash Player) a una política de carga secuencial, a fin de estudiar su impacto en el uso de memoria.

A parte de un mejor uso de la memoria, el límite previo detectado ha valores próximos al 1Gbyte desapareció, permitiendo en pruebas realizadas un uso de 1'6 Gbytes sin detectar ningún problema.

Se especula que el proceso de carga de ficheros que gestiona internamente el Flash Player tiene un límite/*bug* en sus buffer intermedios para múltiples ficheros de gran tamaño.

Resumen

Castellano

Este trabajo desarrolla el proceso de diseño e implementación de una interfaz web que permite la exploración en detalle de las relaciones entre genomas completos.
La interfaz permite la comparación simultánea de nueve genomas, representando en cada gráfica las relaciones entre cada par de genomas junto los genes identificados de cada uno de ellos. Es capaz de trabajar con genomas del dominio Eukaryota y se adapta a la capacidad de cómputo de la máquina cliente.
La información representada son MUMs (*Maximal Unique Matching*, secuencia máxima y única encontrada en ambos genomas) y SuperMUMs (agrupación de MUMs mediante *Approximate String Matching*). Los datos son previamente calculados y accesibles desde un servidor web.

Català

Aquest treball desenvolupa el procés de disseny i implementació d'una interfície web que permet l'exploració detallada de les relacions entre genomes complets.
La interfície permet la comparació simultània de nou genomes, representant en cada gràfica les relacions entre cada parell de genomes junt els gens identificats de cadascun d'ells. És capaç de treballar amb genomes del domini Eukaryota i s'adapta a la capacitat de còmput de la màquina client.
La informació representada són MUMs (*Maximal Unique Matching*, seqüència màxima i única trobada en tots dos genomes) i SuperMUMs (agrupació de MUMs mitjançant *Approximate String Matching*). Les dades són prèviament calculades i accessibles des d'un servidor web.

English

This work develops the process of designing and implementing a web interface that allows detailed examination of the relationship between complete genomes.
The interface allows simultaneous comparison of nine genomes, each graph is representing the relationships between each pair of genomes with the genes identified for each of them. Able to work with domain Eukaryote genomes and adapts to the computing capacity of the client machine.
The information represented are MUMs (Unique Maximal Matching, maximum and unique sequence found in both genomes) and SuperMUMs (MUMs grouping by Approximate String Matching). The data is pre-calculated and accessible from a web server.