APORTACIÓN BIOINFORMÁTICA A LA BIOLOGÍA DE ESPECIES MARINAS

Tesis Doctoral

Hicham Benzekri

2016

Director: Dr. M. Gonzalo Claros

UNIVERSIDAD DE MÁLAGA

AUTOR: Hicham Benzekri

http://orcid.org/0000-0003-1064-5075

EDITA: Publicaciones y Divulgación Científica. Universidad de Málaga

TESIS DOCTORAL

APORTACÓN BIOINFORMÁTICA A LA BIOLOGÍA DE ESPECIES MARINAS

HICHAM BENZEKRI

UNIVERSIDAD DE MÁLAGA
Departamento de Biología Molecular y Bioquímica
Facultad de Ciencias
y
Plataforma Andaluza de Bioinformática
Edificio de Bioinnovación

Málaga. junio de 2016

APORTACIÓN BIOINFORMÁTICA A LA BIOLOGÍA DE ESPECIES MARINAS

Memoria presentada por:

Hicham Benzekri

Para optar al grado de Doctor por la Universidad de Málaga

Tesis realizada bajo la dirección del Dr. M. Gonzalo Claros Díaz en la Plataforma Andaluza de Bioinformática y el Departamento de Biología Molecular y Bioquímica de la Universidad de Málaga

Fdo. Hicham Benzekri

Vº.Bº. DIRECTOR DE LA TESIS DOCTORAL:

Fdo. M. Gonzalo Claros Díaz

Málaga, 11 de noviembre de 2015

D. M. Gonzalo Claros Díaz, Investigador de la Plataforma Andaluza de Bioinformática y el Departamento de Biología Molecular y Bioquímica de la Universidad de Málaga.

CERTIFICA:

Que Don Hicham Benzekri, Ingeniero agronomo, ha realizado bajo mi dirección en el Departamento de Biología Molecular y Bioquímica y en la Plataforma Andaluza de Bioinformática de la Universidad de Málaga, el trabajo de investigación recogido en la presente memoria de Tesis Doctoral que lleva por título: "Aportación bioinformática a la biología de especies marinas".

Tras la revisión de la presente Memoria se ha estimado oportuna su presentación ante la Comisión de Evaluación correspondiente, por lo que autorizo su exposición y defensa para optar al grado de Doctor.

Y para que así conste, en cumplimiento de las disposiciones legales vigentes, firmo el presente certificado.

Málaga, 11 de noviembre de 2015

El Director de la Tesis,

Dr. D. M. Gonzalo Claros Díaz

Agradecimientos

En primer lugar quiero agradecer de manera especial y sincera al Dr. M. Gonzalo Claros Díaz por aceptarme para realizar esta tesis bajo su dirección y por haberme facilitado los medios para poder llevar a cabo este trabajo de investigación. Le agradezco sus atentas y rápidas respuestas a las diferentes dudas e inquietudes surgidas durante el desarrollo de este trabajo, su paciencia, confianza, y todo el apoyo que me ha ofrecido.

Quiero expresar también mi más sincero agradecimiento al Dr. Manuel Manchado Campaña, del grupo de investigación de IFAPA El Toruño (Cádiz), por haber financiado el trabajo de tesis, por su apoyo y confianza, y por haber sido tan atento y dispuesto a resolver todas mis dudas durante el periodo de investigación.

Agradezco al Dr. Oswaldo Trelles Salazar por haberme acogido en su departamento durante un periodo de esta tesis, y a los miembros de su grupo de investigación: Nono, Javier, Maxi, Johan, Alfredo, José Manuel y Vicki, gracias por serme tan útiles en el trabajo y por todos esos momentos agradables que pasamos juntos.

A los compañeros de la plataforma de bioinformática: Rocío, Darío, Rafa, Noé, Rosario, Isa, Pedro y David, muchas gracias por estar presentes y ser tan atentos y amistosos. Nunca olvidaré esos momentos que pasé en este ambiente tan agradable y lleno de aprendizaje y de buen humor. A Rocío gracias por iniciarme en este mundo de la bioinformática y por tu inestimable ayuda y apoyo durante todos estos años. A Noé, gracias por haberme introducido en el trabajo de la tesis y por todo los que me has enseñado; a Darío gracias por ser tan amable ofreciéndome tu ayuda en cuanto la necesitaba y por todo lo que me has enseñado; a Rafa gracias por tu cordialidad y tu ayuda.

A los demás compañeros del Edificio de bioinnovación: Pepi, Diego, Carlos, Xhanti y Rocío gracias por los ratos agradables que pasamos juntos durante las comidas y por vuestra simpatía y afecto.

A mis grandes amigos que conocí en España: Said, Issam, Victor y su novia Luli, y Paco, les agradezco muchísimo su apoyo y el estar tan presentes en mis momentos difíciles.

A mi familia, mis hermanos, mis cuñados, mis sobrinos y en especial a mis padres, ya que sin ellos nada de esto habría sido posible.

A todos los que confiaron en mí y me apoyaron.

Acrónimos y Abreviaturas

ADN:	ácido desoxirribonucleico
ARN:	ácido ribonucleico
ADNc:	copia en ácido ribonucleico
ARNm:	ácido ribonucleico mensajero
ARNnc:	ARN no codificante
ARNr:	ácido ribonucleico ribosómico
ARNt:	ARN de transferencia
API:	**i**nterfaz de programación de aplicaciones
BAC:	cromosoma artificial bacteriano
BLAST:	Basic Local Alignment Search Tool
CPU:	Unidad central de procesamiento
CEG :	*Core Eukaryotic Genes*
dATP:	trifosfato de desoxiadenosina
dCTP:	trifosfato de desoxicitidina
ddNTP:	didesoxinucleótidos trifosfato
DDBJ:	Banco de Datos de ADN de Japón
dGTP:	trifosfato de desoxiguanosina
dTTP:	trifosfato de desoxitimidina
EC:	*Enzyme Commision*
EBI:	European Bioinformatics Institute
EMBL:	Laboratorio Europeo de Biología Molecular
ENA:	European Nucleotide Archive
EST:	etiquetas de secuencias expresadas (del inglés *Expressed Sequence Tag*)
GB:	*gigabyte*
GO:	Gene ontology
HSP:	High-scoring Segment Pairs
HER:	High-entropy region
ID:	identificador
INDEL:	inserciones y deleciones
IPR:	Interpro
JSON:	formato ligero para el intercambio de datos (del inglés *JavaScript object notation*)
KEGG:	Kyoto Encyclopedia of Genes and Genomes
MID:	Molecular Identifier
miRNA:	microARN
MP:	lecturas pareadas de tipo *mate-pair*
NCBI:	National Center for Biotechnology Information
NGS:	**s**ecuenciación de nueva generación (del inglés *next generation sequencing*)
NHGRI:	National Human Genome Research Institute
Nt:	nuleótido
OLC:	*overlap-layout-consensus*
ORF:	Marco abierto de lectura, (del inglés *open reading frame*)
pb:	pares de bases

PCR: reacción en cadena de la polimerasa (del inglés *polymerase chain reaction*)
PE: lecturas pareadas de tipo *paired-end*
RAM: Memoria de acceso aleatorio
ROR: *Ruby On Rails*
SE: lecturas simples
SMRT: *Single Molecule, Real-Time*
SNP: polimorfismos mononucleotídicos
SRA: Sequence Read Archive
SSR: repeticiones de secuencias simples
STS: *sequence-tagged sites*
TB: *terabyte*
UTR: regiones no traducidas de los genes (del inglés *untranslated región*)
VCF: *variant call format*
WGS: *whole genome sequencing*
ZFIN: *Zebrafish Information Network*

Resumen y contextualización

Los recursos marinos son de gran interés económico en todo el mundo. La acuicultura es el sector de producción de alimentos de mayor crecimiento y se espera que en el futuro cercano produzca más alimento para el consumo humano que la pesca de captura. Sin embargo, a diferencia de lo que ha ocurrido en los sectores agrícola, ganadero y avícola donde se han obtenido mejoras en la producción basándose en enfoques modernos de cría sustentados en genómica, las aplicaciones de los principios genéticos para las especies usadas en acuicultura es un campo reciente que ha permitido mejorar el cultivo de sólo unas pocas especies a través de tecnologías clásicas de selección genética como son las cruzas intraespecíficas e interespecíficas.

La aplicación de estrategias de cultivo basadas en biotecnologías genómicas podrían tener un gran potencial ya que permitirán obtener mejoras a largo plazo a través de la identificación de diversos marcadores moleculares que faciliten los programas de cultivo selectivo, el desarrollo de peces transgénicos resistentes a enfermedades y/o con mejores índices de crecimiento, así como la aplicación de diversas técnicas moleculares para el diagnóstico y caracterización de patógenos y el diseño de vacunas moleculares más efectivas, fáciles de producir y de administrar. Así que el uso combinado de las diferentes biotecnologías permitirá potenciar la producción de especies acuícolas.

Este trabajo de doctorado se ha realizado en el grupo de investigación de la Plataforma Andaluza de Bioinformática bajo la dirección del profesor M. Gonzalo Claros Díaz en cooperación el grupo de investigación de grupo de investigación de IFAPA El Toruño (Cádiz) del Dr. Manuel Manchado Campaña. El objetivo final era de realizar un análisis genómico del lenguado senegalés y el lenguado común además de otras especies afines basándonos sobre los datos de lecturas NGS, encaminado hacia la profundización en el conocimiento del transcriptoma y del genoma de estas especies, así como establecer las herramientas bioinformáticas que resultarán más útiles para dichos estudios.

En gran parte, este trabajo ha sido posible gracias al personal y la infraestructura de la Plataforma Andaluza de Bioinformática (PAB), donde es posible acceder a recursos informáticos sin los que parte del trabajo hubiera sido de difícil resolución, además de aprender técnicas de programación, que han permitido que los programas desarrollados en este trabajo puedan ser ejecutados en paralelo.

La financiación para poder acometer esta tarea ha procedido de los siguientes proyectos y contratos:

- Desarrollo de herramientas bioinformáticas para los estudios genómicos y transcriptómicos a partir de datos de secuenciación de lecturas cortas de alto rendimiento para las especies que no tienen un organismo modelo de referencia (NEOGEN) (1/4/2011-30/4/2016; Proyecto de Excelencia de la Junta de Andalucía, P10-CVI-6075). IP: M. G. Claros

- Creación de una base de datos de secuencias obtenidas por 454. Contrato Menor 8.06/5.72.3653 con el IFAPA Centro «El Toruño». 1-4-11 al 30-6-11. IP: M.G. Claros.

- Asistencia técnica para el ensamblaje y anotación de genes implicados en el sistema inmune del lenguado. Contrato Menor 8.06/5.72.3653-1 con el IFAPA Centro «El Toruño». 1-7-11 al 31-10-11. IP: M.G. Claros

- Construcción de la base de datos OysterGeneDB para secuencias de ostras. Contrato Menor 8.06/5.72.3843 con la Universidad de Cádiz, Dpto de Biomedicina, Biotecnología y Salud Pública. 15-6-12 al 14-8-12. IP: M.G. Claros

- Análisis, gestión integración web y desarrollo de las bases de datos genómicas. Contrato 8.06/5.72.3857 con el IFAPA Centro «El Toruño». 13-9-12 al 12-10-13. IP: M. G: Claros

- Implementación de TECnologías INNOvadoras de mejora genética en lenguado senegalés (*Solea senegalensis*) y dorada (*Sparus aurata*) para la optimización de su producción industrial (INNOTECSS) (2014-2017, RTA2013-00023-C02-01, MINECO-INIA). IP: Manuel Manchado Campaña

De este trabajo se han publicado los siguientes capítulos de libro y artículos:

- Benzekri et al. (2014). *De novo* assembly, characterization and functional annotation of Senegalese sole (*Solea senegalensis*) and common sole (*Solea solea*) transcriptomes: integration in a database and design of a microarray. *BMC Genomics* 15, 952. doi: 10.1186/1471-2164-15-952 (ISSN 1471-2164)

- Canales, J; Bautista, R; Label, P; Gomez-Maldonado, J; Lesur, I; Fernández-Pozo, N; Rueda-López, M; Guerrero-Fernández, D; Castro-Rodriguez, V; Benzekri, H; Cañas, R; Guevara, M.A.; Rodrigues, A; Seoane, P; Teyssier, C; Ehrenmann, F; Morel, A; Le Provost, G; Lalanne, C; Noirot, C; Klopp, C; Raymond, I; García-Gutiérrez, A; Trontin, J.F.; Lelu-Walter, M.A.; Miguel, C; Cervera, M.T.; Cantón, F.R.; Plomion, C; Harvengt, L; Avila, C; Claros, M.G.; Cánovas, F (2014) *De novo* assembly of maritime pine transcriptome: implications for forest breeding and biotechnology. *Plant Biotech. J.* 12(3), 286-299 doi: 10.1111/pbi.12136

- X Cousin, M.G. Claros, D Mazurais, R Bautista, H Benzekri, M-L Bégout, M Ponce, P Armesto, J Zambonino, J V Planas, M Manchado (2013) Genome-wide gene expression analysis during Solea sp. embryo-larval development. *Commun Agric Appl Biol Sci.*, 78 (4): 91-2 (ISSN: 1379-1176)

- M.G. Claros, Bautista R., Guerrero-Fernández D., Benzerki H., Seoane P., Fernández-Pozo N. (2014) Why Assembling Plant Genome Sequences Is So Challenging?. En *The Role of Bioinformatics in Agriculture*. Ed. Santosh Kumar; Apple Academic Press Inc, Oakville (Canadá), pp. 27-55 (ISBN 13-978-1-77188-003-9)

- H. Benzekri; D. Guerrero-Fernández; R. Bautista; M.G. Claros (2013) Detecting and correcting mis-assembled reads in contigs. En *Procceedings IWBBIO 2013: International Work-Conference on Bioinformatics and Biomedical Engineering*. Eds. Ortuno, F; Rojas, I. Univ. Granada, Granada, pp. 345-351 (ISBN 978-84-15814-13-9)

- A. Muñoz-Mérida, J. Ríos, H. Benzekri, O Trelles (2012). Statistical significance for NGS reads similarities, *Bioinformatics for Personalized Medicine*, 1-7

Y se han presentado las siguientes comunicaciones a congresos:

- Manchado M., Benzekri H., Juan J Sánchez, Bautista, R., Cousin, X., Planas J.V., Rebordinos L, Claros, M.G. Development of genomic tools in senegalese sole: transcriptome assembly, annotated database, microarray and genome draft. ISGA XII - The International Symposium on Genetics in Aquaculture XII. Santiago de Compostela, 21-27/6/15. INT

- Hicham Benzekri, Pedro Seoane, Rosario Carmona, Rocío Bautista, Darío Guerrero-Fernández, Noé Fernández- Pozo, M.G. Claros. A workflow with dedicated tools for preparing reference transcriptomes from non-model organisms has evidenced important biological information. XII Symposium on Bioinformatics. Sevilla, 21-24/09/14. INT

- Hicham Benzekri, Darío Guerrero-Fernández, Rocío Bautista, and M.G. Claros. "Detecting and correcting mis-assembled reads in contigs". International Work-Conference on Bioinformatics and Biomedical Engineering IWBBIO13. Granada, marzo 2013. INT

- Hicham Benzekri, Rocío Bautista, Darío Guerrero-Fernández, Noé Fernández-Pozo, M. G Claros. «A reliable pipeline for a transcriptome reference in non-model species». International Conference The next NGS Challenge: data processing and integration. Valencia, mayo 2013. INT

- M. Ponce, C. Berbel, M. Aparicio, H. Benzekri, M.G. Claros, M. Manchado. «Secuenciación del genoma de *Photobacterium damselae.* Estudios in silico para la identificación de genes implicados en la patogenicidad». AQUAGENET, cooperation and biotechnology for a sustainable aquaculture in SUDOE region. Puerto de Santa María, diciembre 2013. INT

- H. Benzekri, N. Fernández-Pozo, D. Guerrero-Fernández, R. Bautista, M.G. Claros "Aproximación bioinformática al transcriptoma de Solea y su disponibilidad en SoleaDB". Biotecnología y recursos genómicos aplicados a la acuicultura. Avances logrados en AQUAGENET. Puerto Real, mayo 2012. INT

- H. Benzekri, A. Muñoz-Mérida, M.G. Claros and O. Trelles "Rapid and efficient contigs mapping and ordering with ICMapper, Student symposium XI Jornadas de Bioinformática (JBI2012). Barcelona, enero 2012. INT.

Indice general

Parte I
Introducción

I. Introducción

I.1. La importancia de la secuenciación a día de hoy

La secuenciación del ADN es el proceso de determinación del orden preciso de nucleótidos en una molécula de ADN. Incluye a todas los métodos y tecnologías utilizadas para determinar el orden de cuatro bases: adenina, guanina, citosina y tiamina. Determinar la secuencia de ADN es útil en el estudio de la investigación básica de los procesos biológicos fundamentales.

La genética molecular ha acelerado significativamente la investigación y los descubrimientos en biología y causaron una revolución científica en varios campos:

- En el campo de la investigación médica, el diagnostico de muchas enfermedades es mucho más eficaz. Se puede evaluar la susceptibilidad de las personas a enfermedades específicas y suministrar los medicamentos más adecuados contra estas enfermedades.
- En el campo legal, las pruebas de paternidad y forenses son más precisas con las tecnologías de la PCR y secuenciación.
- En ganadería y agricultura está facilitando la mejora de los animales y vegetales en relación con su capacidad productiva o resistencia a las enfermedades.

Las técnicas de secuenciación de ADN han evolucionado mucho a lo largo de los últimos años y han generado un gran impacto sobre la investigación científica aplicada a la biología y a la medicina, donde se ha pasado de secuenciar los genes uno a uno de modo manual en los años 80 [1, 2] a proyectos de secuenciación de mil o más genomas en la actualidad [3-5]. En los siguientes apartados, vamos a ver cómo se ha pasado de la secuenciación manual de varios cientos de nucleótidos a la secuenciación automática con capilares y posteriormente al desarrollo de la secuenciación de nueva generación (NGS, del inglés Next Generation Sequencing). Como la NGS genera secuencias de 75 a 800 nt de uno o varios millones de moléculas a la vez en reacciones a escala nanotecnológica, producen un volumen de secuencias tal que su procesamiento necesita la ayuda de la bioinformática.

La secuenciación de los organismos representa la primera etapa del proceso de obtención de sus transcriptomas o genomas anotados (figura I.1)

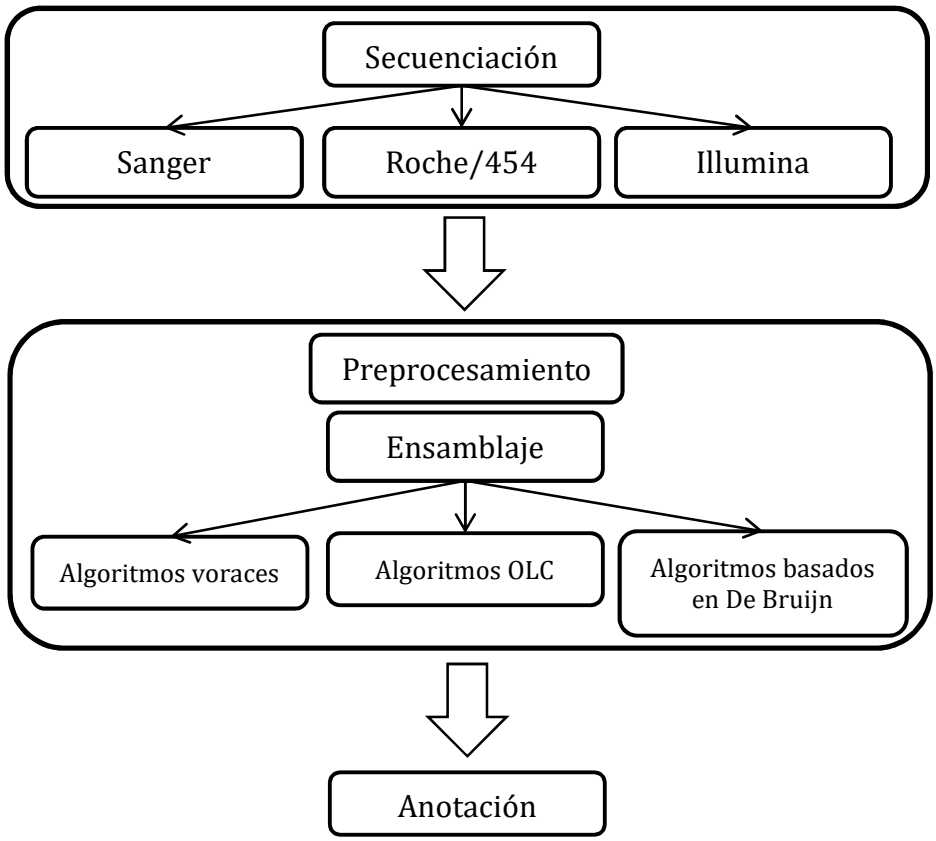

Figura I.1: *Esquema general del flujo de trabajo para obtener genomas o transcriptomas anotados*

I.1.1. Secuenciación de tipo Sanger, hoy considerado "clásico".

En 1977 se publicaron a la vez el método de secuenciación química de Maxan y Gilbert [6] y el método de secuenciación de terminación de cadena de Sanger [7], también conocido como método enzimático o secuenciación por didesoxinucleótidos. Este método, esquematizado en la figura I.2, se basa en la formación de un extremo 3' terminador mediante la incorporación de didesoxinucleótidos trifosfato (ddNTP) por lo que la ADN polimerasa no puede seguir extendiendo al carecer de grupo hidroxilo en 3'. La ADN polimerasa sintetiza hebras complementarias a la que se quiere secuenciar, por lo tanto necesita un cebador, un oligonucleótido diseñado para que se hibride con el extremo 3' de ésta, además de una mezcla de dNTP (uno de ellos radiactivo) y ddNTP. Para llevar a cabo la secuenciación se realizan cuatro reacciones por separado, cada una con un ddNTP diferente. De este modo se obtienen secuencias truncadas en diferentes posiciones de 3' del nucleótido correspondiente a cada tubo. Estas moléculas se separaran por tamaño en una electroforesis, utilizando un carril para cada tubo, para visualizar el patrón de bandas del que se puede deducir la secuencia [7].

Frederick Sanger fue premiado con su segundo Nobel de química en 1980 por su contribución en la determinación de las secuencias de los ácidos nucleicos, compartiendo el premio con Walter Gilbert y Paul Berg [8], lo que puede dar una idea de lo que su técnica

aportó a la investigación científica, posibilitando la secuenciación, entre otros, del genoma humano [9, 10].

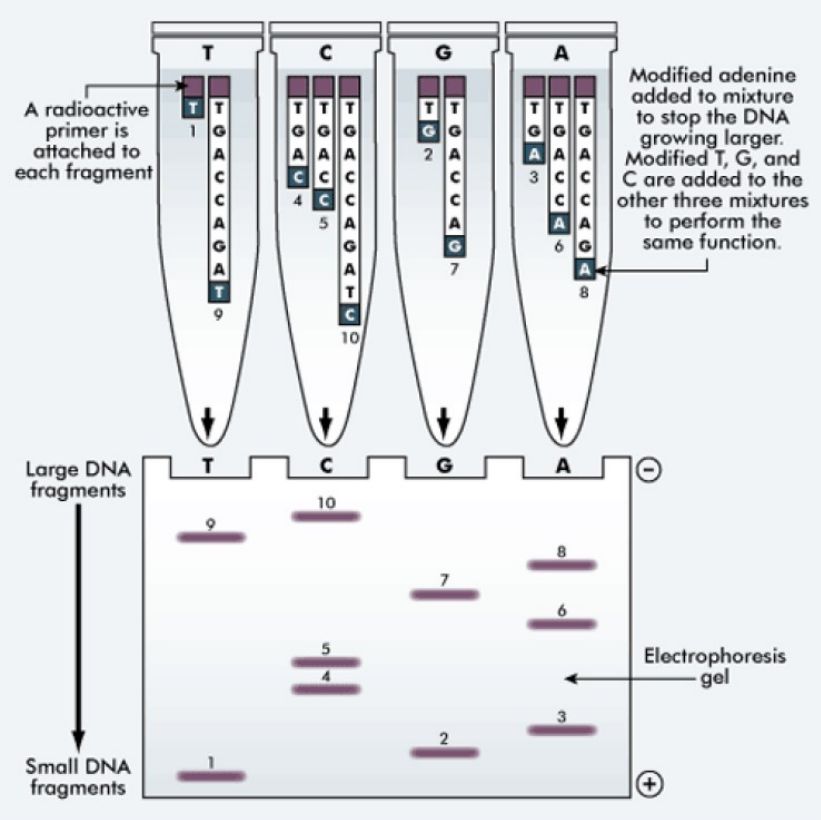

Figura I.2: *(tomada de* `http://mol-biol4masters.masters.grkraj.org`*) Método de secuenciación por didesoxinucleótidos de Sanger*

El método de Sanger se adaptó posteriormente para la secuenciación automática en capilares, sustituyendo el marcaje con isótopos radiactivos por un marcaje con 4 fluorocromos diferentes. Con ello se consiguió realizar la reacción en un único tubo y migrarla en un solo capilar (figura I.3-a), años más tarde las máquinas pasaron a incorporar varios capilares (de 4 a 384) para paralelizar este proceso. Después de tres décadas de continuo perfeccionamiento, es posible obtener secuencias de unas 1000 pb con una fiabilidad del 99,999 % y con un coste de unos 0,5 $ por kilobase (Figura I.4).

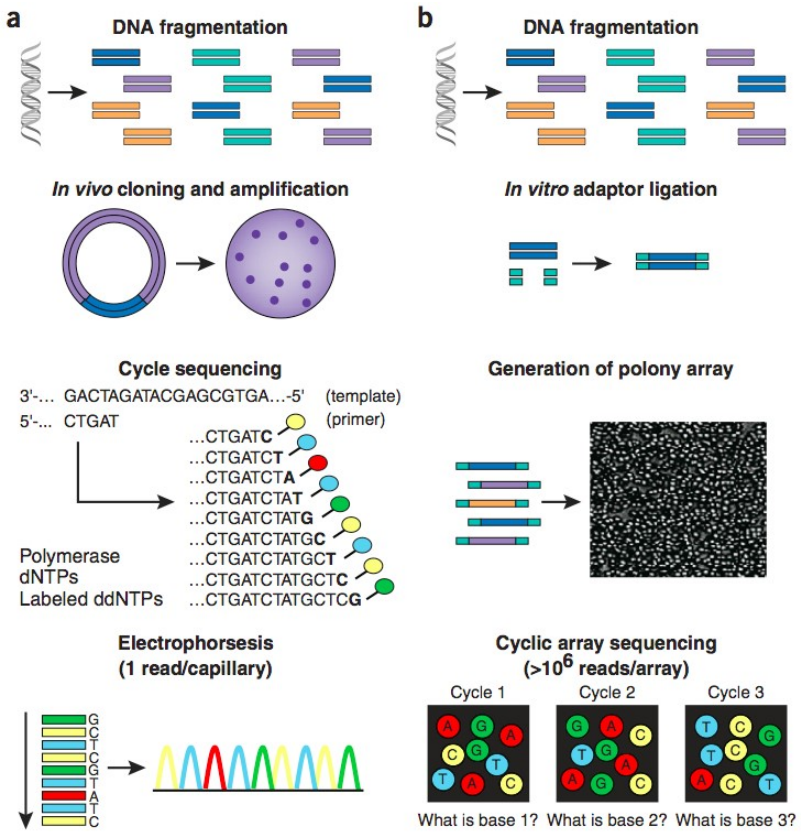

Figura I.3: *(tomada de [11]) Paralelismo de las etapas a seguir en la secuenciación automática basada en el método de Sanger y en las nuevas tecnologías de secuenciación de alto rendimiento. (a) Secuenciación automática Sanger: las etapas se pueden resumir en fragmentación del ADN, preparación de las muestras in vivo, elongación de los cebadores, ordenamiento de los fragmentos por electroforesis en un capilar y lectura de la señal por un receptor. (b) Secuenciación de nueva generación (NGS): sus etapas son fragmentación del ADN, preparación de las muestras in vitro, elongación de los cebadores y captura de imágenes de la micromatriz*

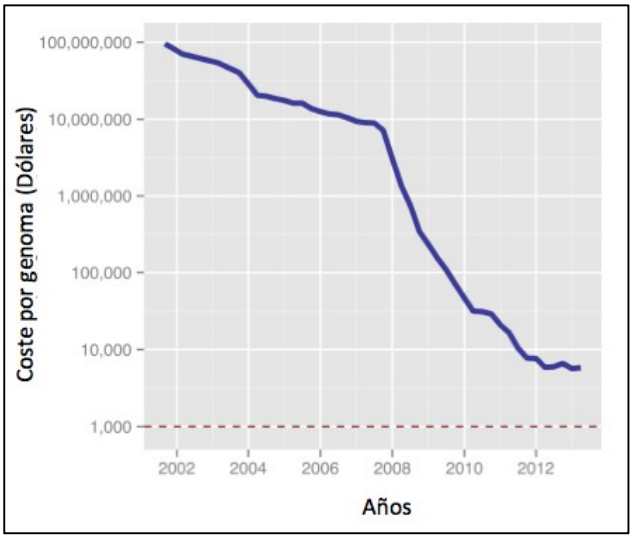

Figura I.4: *Coste de secuenciación de genomas estimado por NHGRI (National Human Genome Research Institute) (Septiembre 2001 – Abril 2013)*

I.1.2. Plataforma 454 Roche

En 2005 Roche presenta el primer método comercial de la secuenciación de nueva generación (NGS), un sistema de secuenciación escalable, verdaderamente paralelizable (no como las 96 o 384 reacciones en paralelo máximas de la secuenciación automática), y de mayor capacidad que los de tipo Sanger. Este método se basa en chips de fibra óptica de 60 x 60 mm2 que contienen aproximadamente 1 600 000 micropocillos (Picotiter plate), en los que entra una microesfera que contiene todo lo necesario para una PCR y una molécula de ADN a secuenciar. Con la PCR se amplifica la molécula a secuenciar, y luego, en la reacción de secuenciación, se añade un nucleótido diferente a cada vez. Si es el nucleótido que necesita la cadena de ADN en formación, se emitirá luz por la acción de la luciferasa (figura I.5). A cada paso se añade un nucleótido diferente y se toma una imagen fotográfica de todo el chip. En función de los pocillos que emitieron luz al añadir cada nucleótido se obtiene la secuencia del ADN que hay en cada micropocillo. De esta forma se consigue secuenciar 25 millones de bases con una precisión del 99% en 4 horas [12].

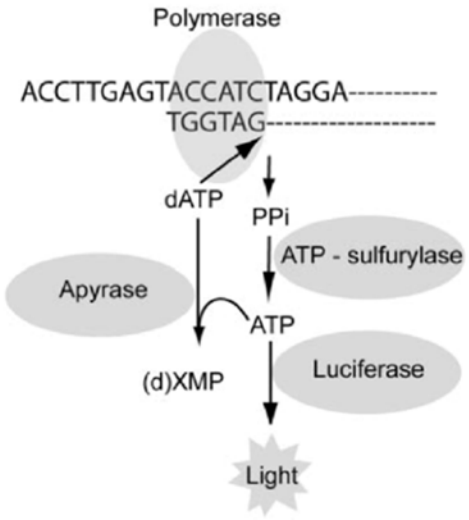

Figura I.5: *(tomada de [13]) Enzimas acopladas en la polimerización del ADN que intervienen en la pirosecuenciación*

En la figura I.3-b aparecen las etapas clave de este método, que se resumen a continuación:

- Fragmentación física del ADN por nebulización.
- Preparación *in vitro* de las muestras de manera muy sencilla, en la que solo se requiere la ligación de adaptadores a los fragmentos de ADN, sin clonación.
- Elongación de los cebadores en cada uno de los millones de micropocillos en los que se está secuenciando una molécula. En cada paso se añade un desoxinucleótido diferente (dATP, dCTP, dGTP, dTTP) y cada vez que alguno se incorpora a la cadena se

libera un pirofosfato, que será sustrato de varias reacciones enzimáticas acopladas (figura I.3), en las que la cantidad de pirofosfato liberado se mide en forma de intensidad luminosa.

- Captura de imágenes de la matriz después de cada etapa de reacción con una cámara CCD, la cual utiliza una pequeña pieza rectangular de silicio en lugar de un trozo de película para recibir la luz entrante. Con la información obtenida en este paso se obtienen los flujogramas (también denominados pirogramas) que determinarán la secuencia (figura I.6).

El método 454 de Roche es el más costoso de los de nueva generación, pero su ventaja es que además de ser eficiente (con los kits modernos de GS FLX Titanium es posible de generar 1 millón de lecturas en 23 horas [http://454.com/downloads/GSFLXApplicationFlyer_FINALv2.pdf]) producen lecturas de mayor longitud, hoy en día muy cercana a la que proporciona la tecnología de Sanger. Las lecturas largas son especialmente útiles cuando se estudian organismos con genomas complejos con un alto número de repeticiones, ya que cuanto más larga sea la secuencia, mayor será la probabilidad de que contenga una repetición corta completa, y no se generarán huecos al ensamblarlas. Así pues, por el hecho de ser la técnica de NGS que produce lecturas de mayor longitud, es la que más se utiliza para el estudio de genomas [14-18] y transcriptomas de organismos no modelo [19-25], o cualquier tipo de ensamblaje *de novo*.

I.1.3. Plataformas Illumina y Solid

La técnica de secuenciación de Illumina tiene su origen en las ideas de Shankar Balasubramanian y David Klenerman de los años 90 [26], y en el trabajo hecho por Turcatti et al. [27, 28]. Se denomina también el método de "secuenciación por síntesis" que a diferencia de la técnica de secuenciación Roche 454, utiliza la química de terminación reversible de nucleótidos análogos [29].

La compañía Solexa fue fundada en 1998. Después de 8 años de investigación y desarrollo, la fusión en 2004 con la compañía Manteia y en 2005 con la compañía de instrumentación Lynx Therapeuticsm, en 2006 Solexa pudo lanzar en el mercado su primer secuenciador ADN, Genome Analyser. Un año después, Solexa fue adquirida por Illumina.

La figura I.7 muestra una vista general del flujo de trabajo de secuenciación de Illumina. Como en la técnica de secuenciación de Roche 454, la construcción de librerías se realiza a través del fraccionamiento de ADN utilizando la nebulización, corrección de los extremos y ligación de adaptadores utilizando enzimas. Pero ahora, la amplificación por PCR del ADN se produce sobre un sustrato solido (flowcell) [28, 30]. Aquí, los fragmentos de ADN flanqueados por un adaptador se unen a una superficie cubierta de oligonucleótidos. La alteración de los ciclos de extensión de la polimerasa bst y la desnaturalización con formamida crea copias de la plantilla del fragmento ADN. La inmovilización asegura que

todos los amplicones procedentes de la plantilla de molécula única se agrupen en la superficie. Cada grupo contiene 1000 copias clónales de la misma plantilla.

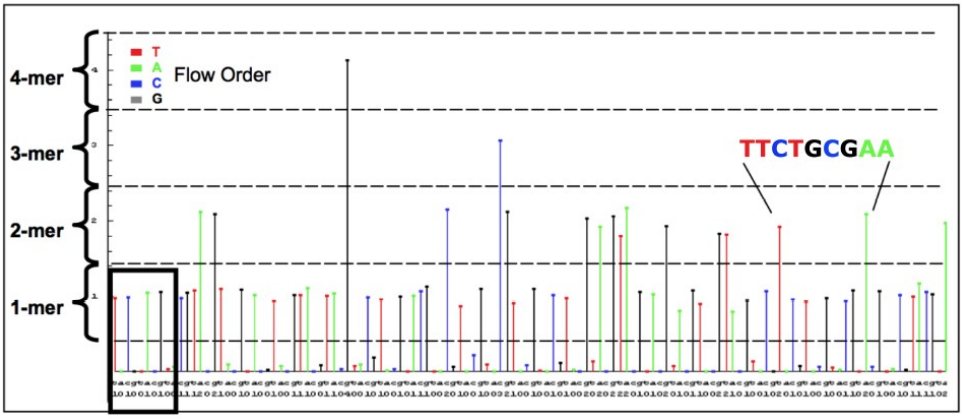

Figura I.6: *(tomada de* `http://www.454.com/downloads/news-events/how-genome-sequencing-is-done_FINAL.pdf` *) Flujograma de una reacción con 454/FLX. En cada ciclo de secuenciación se añade un nuevo desoxinucleótido en el orden TACG. Cuando el desoxinucleótido añadido se incorpora a la cadena, en el flujograma se indica con una barra, que variará su altura en función del número de desoxinucleótidos incorporados y su color en función del desoxinucleótido añadido en ese ciclo. En el recuadro negro se señala la clave de la reacción, 4 nucleótidos introducidos al principio de cada secuencia para calibrar la intensidad de luz detectada por nucléotido. Finalmente, un programa informático determina la secuencia basándose en la altura de cada emisión de luz en el flujograma*

Las 8 secciones de la placa permiten la secuenciación paralela de 8 librerías independientes (figura I.7). Después de la generación de grupos (clusters), un cebador de secuenciación se hibrida con uno de los adaptadores flanqueantes del fragmento de ADN de interés. En cada ciclo se incorpora una única base con nucleótidos modificados utilizando una ADN polimerasa modificada [29]. Un grupo de bloqueo 3'-O-azidomethyl garantiza que se incorpore una base única, y una de las cuatro etiquetas permite la detección de las diferentes bases de ADN [27, 29]. Después de la adquisición de las cuatro imágenes en diferentes canales, el ciclo de secuenciación termina con una escisión química del fluoróforo y del grupo de bloqueo, permitiendo la incorporación de base en el ciclo siguiente. Por último, con el procesamiento de las imágenes y la filtración de lecturas de baja calidad se genera la salida final en ficheros de secuencias en formato FASTQ que es un formato basado en texto que permite almacenar tanto la secuencia de los nucleótidos como sus escores de calidad.

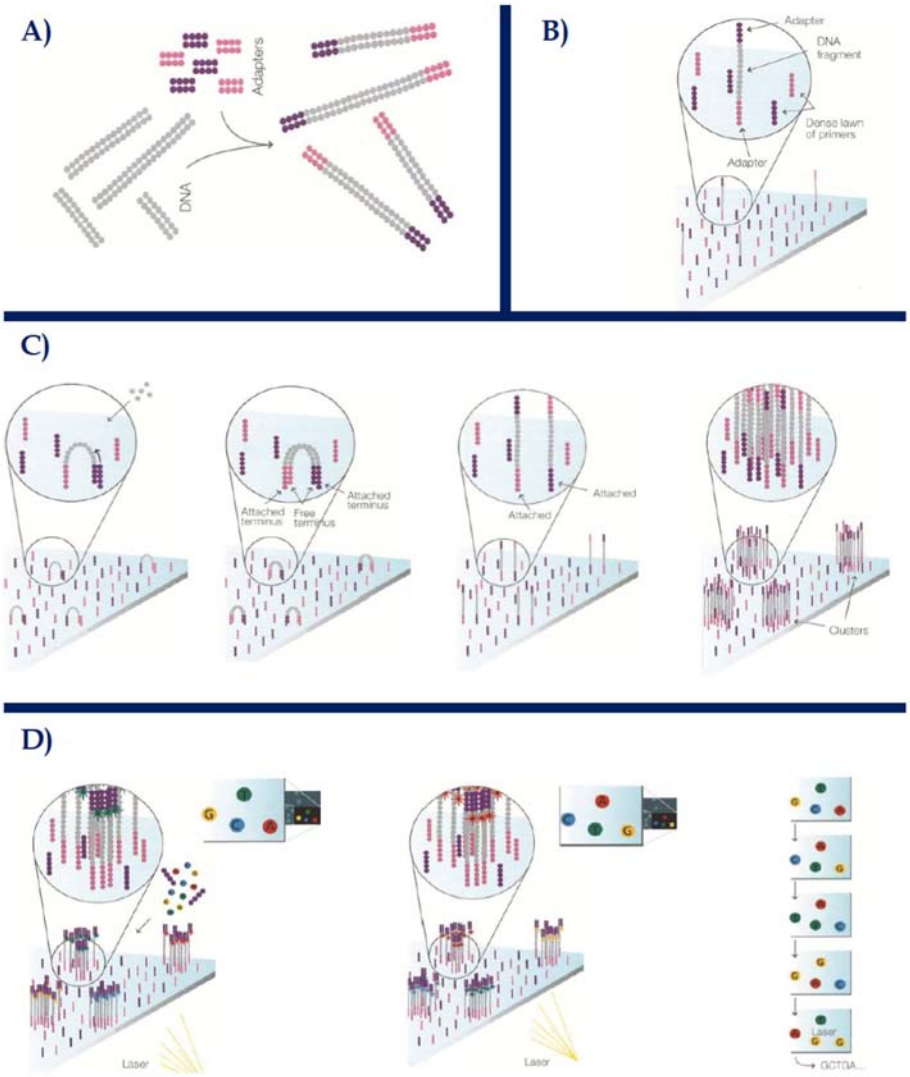

Figura I.7: *Flujo de trabajo de la secuenciación Illumina. Incluye las mismas grandes etapas que en el caso de la secuenciación 454. A) Construcción de la librería fraccionando el ADN por la nebulización, y posteriormente hacer romos los extremos y ligación del adaptador. B) Los adaptadores que flanquean los fragmentos de ADN se unen con una superficie cubierta de oligonucleótidos C) La alteración de los ciclos de extensión de la polimerasa bst y la desnaturalización utilizando la formamida crea grupos de 1000 copias clónales de la plantilla. D) Inicio de la secuenciación por hibridación de un primer de secuenciación. En cada ciclo, se incorpora una base única con nucleótidos modificados utilizando una ADN polimerasa modificada. Una de las cuatro etiquetas permite la detección de las diferentes bases de ADN. Después de la adquisición de cuatro imágenes en diferentes canales, el ciclo de secuenciación termina con una escisión química del fluofóro y el grupo de bloqueo, permitiendo la incorporación de base en el siguiente ciclo. Posteriormente, las imágenes se convierten a fichero de secuencias a través del análisis de imagen, base calling, y la filtración de secuencias*

Con la última versión de esta línea de secuenciadores el HiSeq 3000/HiSeq 4000, es posible secuenciar más de 400 gigabases al día (HiSeq 4000) lo que corresponde a 1 terabase (1,4 Tb) de datos por run (ejecución) de 3,5 días y una longitud de lecturas de 2 x 150 (`http://www.illumina.com/systems/hiseq-3000-4000.html`).

I.1.4. Plataformas de secuenciación masiva de tercera generación

Un hilo común entre las tecnologías de secuenciación en los últimos años ha sido la mejora continua del rendimiento, lo que implica el incremento del número y longitud de lecturas en una misma carrera con la consecuente reducción de costes por base secuenciada.

Se han desarrollado varias tecnologías, que están en constante evolución, cuyo objetivo es obtener secuencias más largas y de mayor calidad para reducir la dificultad de los ensamblajes. Entre estas tecnologías es posible encontrar SMRT (Pacific Biosciences, Menlo Park, USA), y Helicos (Helicos Biosciences, Cambridge, USA) [31]. Además, en el caso de SMRT de Pacific Biosciences se requiere menos cantidad de fungibles que en otras tecnologías de NGS, por lo que se espera que en un futuro se reduzcan más los costes de secuenciación (figura I.4).

I.1.5. Técnica de secuenciación mediante lecturas pareadas

La técnica de secuenciación mediante lecturas pareadas fue descrita por Edwards y Caskey en 1991 [32], y posteriormente varios grupos desarrollaron variantes de esta técnica [33]. Se basa en obtener las secuencias de los extremos de una molécula de ADN de longitud conocida. En el proceso de ensamblaje, las secuencias pareadas contienen la misma información que las secuencias simples y además se conoce también qué posición relativa debe tener con respecto a su pareja. Esta información se utiliza posteriormente para corregir errores de ensamblaje, para reconocer la presencia de repeticiones largas, y para generar supercontigs (también, denominados *scaffolds*; ver apartado I.2.2).

Existen varios protocolos de generación de secuencias pareadas en función de la longitud del inserto que se quiera secuenciar y del tipo de secuenciación. En el caso de Illumina, se obtienen (a) extremos emparejados (paired-ends [PE], para insertos de hasta 800 pb) y (b) parejas conjugadas (mate-pairs [MP], para insertos de al menos 3 kb o más) (figura I.8). En el caso de los extremos emparejados, los fragmentos son directamente ligados a los adaptadores para su posterior proceso de secuenciación. En el caso de las parejas conjugadas el proceso es más complejo. Los fragmentos (usualmente de varios kb) se circularizan, luego se rompe la unión formada por los extremos, y finalmente se secuencian los fragmentos resultantes.

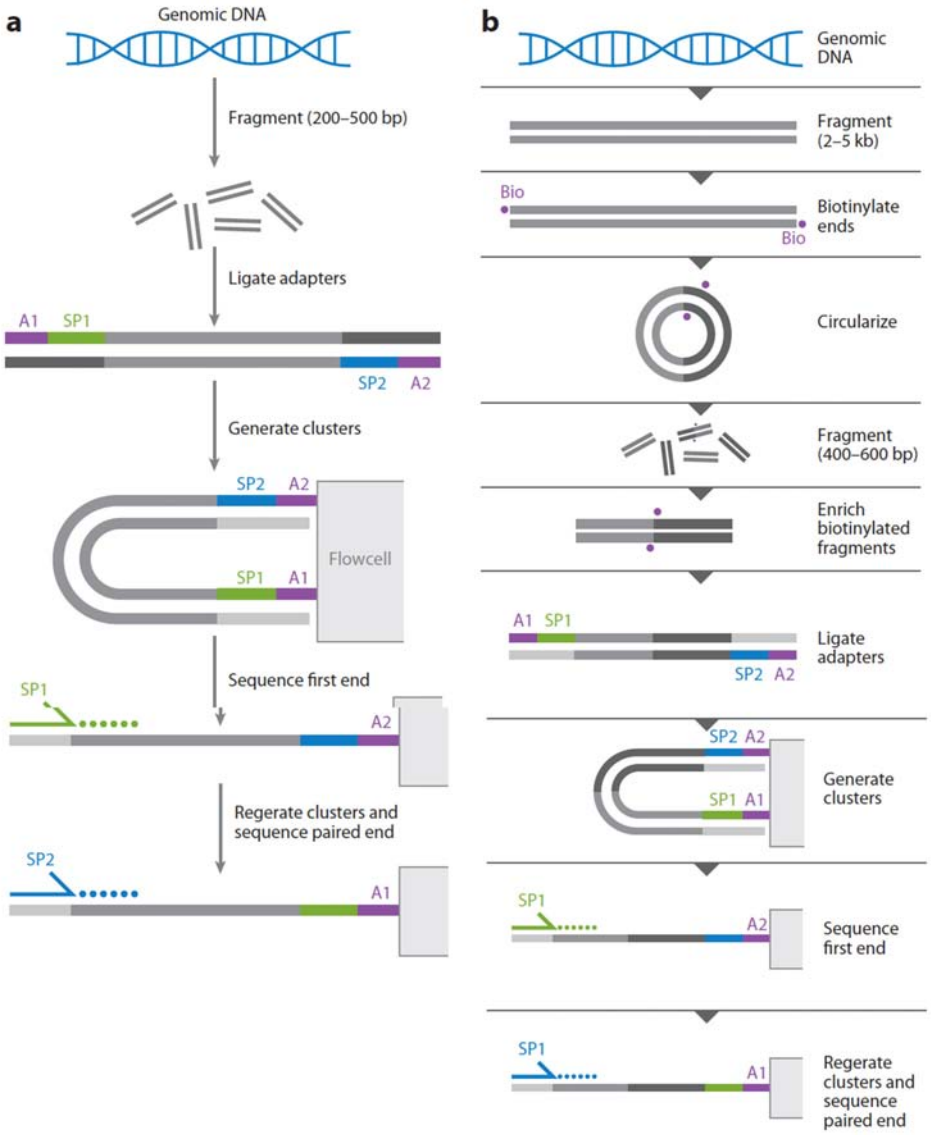

Figura I.8: *(tomada de [34]) Comparación entre las técnicas de secuenciación (a) extremos emparejados (paired-end) y (b) parejas conjugadas (mate-pair)*

I.2. Reconstrucción de fragmentos de secuencias

I.2.1. Preprocesamiento.

Después de obtener las lecturas de una secuenciación es indispensable preprocesarlas para eliminar los elementos que provienen de la manipulación del ADN o del ARN en el laboratorio. Así se obtendrá únicamente la parte útil y se descartan los fragmentos de baja calidad, contaminantes, vectores, adaptadores [35], y otros artefactos. En las nuevas tecnologías, además, es posible encontrar otros elementos nuevos que se deben considerar en el preprocesamiento. Por ejemplo, el etiquetado con MID (Roche) u otras etiquetas implica que hay que retirarlas de la secuencia final, y además hay que usarlas para agrupar las lecturas por experimentos. En resumen, la calidad y la fiabilidad del posterior ensamblaje

dependerá de un buen preprocesamiento [36] ya que, de otra manera, los consensos obtenidos podrían contener numerosos errores.

Ya existen programas para preprocesar secuencias de tipo Sanger, como SeqClean (`http://compbio.dfci.harvard.edu/tgi/software/`), Lucy [37], ESTPrep [38] y SeqTrim [36]. Algunas herramientas bioinformáticas como TagCleaner [39] y NovoBarCode (Novocraft) se han diseñado para identificar los MID y otras etiquetas, y clasificar las secuencias procedentes de los diferentes experimentos. En Galaxy [40] (`https://usegalaxy.org`) se puede acceder a distintas herramientas, como por ejemplo FastQC, Groomer, Splitter, etc., para montar un flujo de trabajo de limpieza.

En cuanto a los contaminantes, los más comunes en el laboratorio suelen ser hongos y bacterias, sobre todo procedentes de la rizosfera [41]. También se consideran contaminantes los restos de tejidos humanos procedentes de los investigadores y microorganismos utilizados con frecuencia en el laboratorio, como *E.coli* [42] y *Agrobacterium tumefaciens* [43], este último sobre todo cuando se trabaja con plantas. Cuando se trata de la secuenciación organismos animales, también se consideran contaminantes los organismos que se utilizan como alimentación y/o las especies de bacterias que normalmente infectan al organismo estudiado. Si no se eliminan las secuencias procedentes de organismos contaminantes se corre el riesgo de considerarlas parte del organismo que se está estudiando [44]. Para la detección y eliminación de secuencias contaminantes existen programas como DeconSeq [45].

Ninguno de los programas anteriores se basta por sí solo para un buen preprocesamiento. Por eso, para la NGS se desarrollaron nuevas herramientas bioinformáticas especializadas como SeqTrim-Next [46] o el paquete Galaxy [40].

Después del preprocesamiento de las lecturas viene la etapa de su ensamblaje en secuencias más largas. Existen varios algoritmos de ensamblaje:

I.2.2. Conceptos básicos sobre ensamblaje de secuencias

Un ensamblaje (o montaje) de secuencias se refiere al alineamiento y mezcla de multiples fragmentos de una secuencia de ADN mucho mayor para reconstruir la secuencia original. En el ensamblaje las lecturas se agrupan en contigs y los contigs en scaffolds. Los contigs representan el alineamiento multiple de lecturas (o fragmentos de lecturas). Los scaffolds (a veces llamados supercontigs o metacontigs) definen el orden y la orientación de los contigs y las longitudes de los huecos entre los contigs (figura I.9).

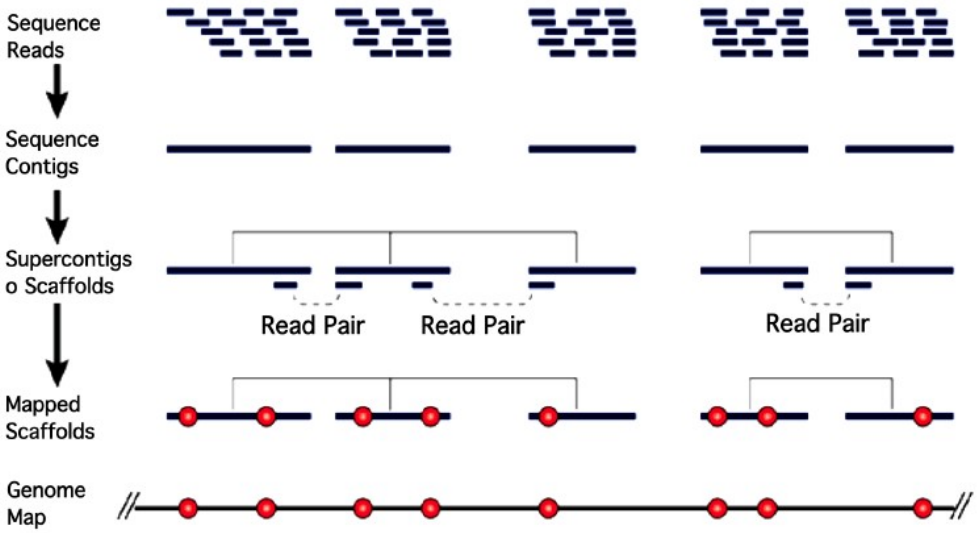

Figura I.9: *(tomada de [47]) Estrategia de secuenciación utilizando secuencias pareadas. A partir de las lecturas generadas en un proyecto de secuenciación de cualquier tecnología o estrategia, se obtienen las secuencias consenso de los contigs mediante el ensamblaje con un programa informático. Las secuencias de los contigs se organizan entonces formando un scaffold basándose en la información de las secuencias pareadas para ordenar contigs no solapantes. Finalmente los scaffold pueden ordenarse igual que el genoma identificando en ellos marcadores que se conozcan en el genoma, como por ejemplo STS, marcadores moleculares y genes (círculos rojos)*

Existen dos tipos principales de ensamblaje:

Ensamblaje comparativo (por mapeo)

Basándose en un genoma secuenciado previamente y que suponemos sea similar al que se quiere ensamblar. El procedimiento básico tratará de colocar cada una de las lecturas en la posición adecuada utilizando el genoma de referencia como guía.

Ensamblaje *de novo*

Intenta reconstruir la secuencia de ADN completa a partir de las lecturas sin ningún tipo de conocimiento previo acerca del genoma a ensamblar. Busca lecturas cuyo final coincida con el principio de otra de forma que se puedan unir para formar fragmentos mayores hasta completar el genoma.

I.2.3. Algoritmos voraces (*greedy*)

Los primeros ensambladores de NGS utilizaban algoritmos voraces [48, 49] que aplican heurísticas en cada fase del algoritmo mediante la cuales pretenden buscar soluciones parciales óptimas y una operación básica: dada una lectura o contig, añade una lectura o

contig más. Esta operación se repite hasta que no sea posible. Cada operación utiliza el solapamiento con más puntuación para generar la siguiente unión. La función de puntuación mide, entre otros indicadores, el número de bases coincidentes en el solapamiento. De esta manera, los contigs crecen por extensión tomando la lectura que se encuentra por el siguiente solapamiento con más puntuación. Los algoritmos voraces pueden atascarse cuando un contig puntual toma lecturas que deben ser de otros contigs.

Los algoritmos voraces son implícitamente algoritmos de grafos ya que simplifican drásticamente el grafo al considerar sólo los enlaces con más puntuación, como optimización pueden instanciar sólo un solapamiento por cada lectura que examinan y pueden también descartar cada solapamiento inmediatamente después de la extensión del contig.

Al igual que todos los ensambladores, los algoritmos voraces necesitan mecanismos para evitar incorporar solapamientos falsos en los contigs, un ensamblador que se base en solapamientos falsos unirá secuencias no relacionadas a cualquier lado de una repetición.

El primer ensamblador descrito para las lecturas cortas fue SSAKE [50] que se diseñó para lecturas no pareadas de longitud uniforme. SHARCGS [51] también trabaja con longitudes uniformes, alta cobertura y lecturas cortas no pareadas, pero añade una funcionalidad de pre- y posprocesado a SSAKE. El preprocesado filtra lecturas erróneas mediante comprobación de un número mínimo de coincidencias exactas de máxima longitud en otras lecturas. VCAKE [52] es otro algoritmo de extensión iterativo distinto a SSAKE y a SHARCGS, puede incorporar coincidencias imperfectas durante la ampliación de los contigs. VCAKE se utilizó en combinación con NEWBLER [12] en una integración para datos híbridos de Solexa y 454 [53]. Otra integración combinó NEWBLER y el CELERA ASSEMBLER [10, 54] para datos híbridos de 454 y Sanger [55]. Ambas integraciones rompen los contigs del primer ensamblador para producir pseudolecturas adecuadas para el segundo ensamblador. Esta última integración ajusta la cobertura de lectura y los indicadores de calidad en las pseudolecturas que genera, lo cual ayuda al segundo ensamblador a dar un peso a los contigs con gran cobertura desde el primer ensamblaje.

I.2.4. Algoritmos de solapamiento-composición-consenso (OLC).

La estrategia OLC (del inglés *overlap-layout-consensus*) fue ampliamente utilizada en los ensambladores para datos Sanger y fue optimizada para genomas grandes en diversos tipos de software incluyendo el CELERA ASSEMBLER [10, 54], ARACHNE [56] y CAP y PCAP [57], EDENA, CABOG, TIGR, ATLAS, PHRAP, PHUSION y SHORTY, ha sido estudiada ampliamente [58].

Los ensambladores que utilizan esta estrategia están muy orientados a ensamblajes *de novo*, utilizan un grafo de solapamientos y operan ejecutando tres fases:

(i). Búsqueda de solapamientos mediante la comparación de lecturas pareadas todas contra todas. El software precalcula los k-meros contenidos en todas las lecturas, selecciona

los solapamientos candidatos que comparten dichos K-meros y calcula las alineaciones utilizando un K-mero como semilla. El descubrimiento de solapamientos es sensible al tamaño de los K-meros, a la longitud mínima de solapamiento y al porcentaje mínimo de identidad requerido en dichos solapamientos. El comportamiento de estos parámetros se ve afectado por los errores de secuenciación y una baja cobertura. Valores mayores conducen a una mayor precisión pero con contigs más cortos.

(ii). La construcción y manipulación de un grafo de solapamientos conduce a un diseño del grafo basado en un conjunto de lecturas representativo, es decir, el grafo no necesita incluir todas las secuencias base, por lo que estos grafos pueden utilizarse en grandes genomas ya que pueden ajustar el tamaño del grafo a la cantidad de memoria que utilizan.

(iii). Generación de la secuencia consenso mediante un alineamiento múltiple de lecturas, aunque no haya un método eficiente para calcularla [59]. Esta fase se puede ejecutar en paralelo, por contigs.

Existen dos ensambladores que aplican la estrategia OLC a lecturas cortas de las plataformas Illumina y SOLiD: el software EDENA [60], que fue desarrollado para lecturas no pareadas de longitud uniforme, descarta lecturas duplicadas y busca todos los solapamientos perfectos libres de errores. Elimina solapamientos individuales que son redundantes con otros solapamientos mediante una aplicación del algoritmo de reducción de solapamientos transitivos [61]. El otro software es SHORTY [62] que trata el caso especial donde unas pocas lecturas pueden actuar como semillas para conseguir lecturas cortas y sus extremos emparejados, y mediante iteraciones utiliza contigs como semillas para generar nuevos contigs.

I.2.5. Algoritmos basados en grafos de De Bruijn

Esta tercera aproximación para ensamblaje de secuencias se utiliza principalmente para lecturas cortas de las plataformas de SOLiD e Illumina. Se basa en grafos de frecuencias de K-meros para tratar grandes cantidades de lecturas cortas, ya que los grafos de K-meros no requieren el análisis de todos los solapamientos mediante la comprobación de todos contra todos, ni tampoco es necesario almacenar las lecturas individuales en sus solapamientos, ni tampoco comprime las secuencias repetitivas. Por el contrario, los grafos de K-meros no mantienen secuencias originales y son grandes consumidores de memoria para grandes genomas, aunque con los sistemas de memoria distribuida se comportan bastante bien [63]. Al final, el grafo de De Bruijn hay que resolverlo con una estrategia euleriana (figura I.10) que se basa en las siguientes etapas:

(i). Fragmentación de las lecturas secuenciadas en una colección de oligomeros (K-meros), donde todas las lecturas tienen la misma longitud (k). Los cebadores redundantes se comprimen en uno solo para reducir la complejidad del análisis, aunque se conserva la información del número de veces que se repiten. Los valores de k suelen asignarse entre 19 y la longitud de las lecturas; en la práctica, estos valores suelen escogerse basándose en un

experimento previo con datos similares [64]. Esta fragmentación de lecturas evita el paso limitante de tener que comparar todas las secuencias entre sí.

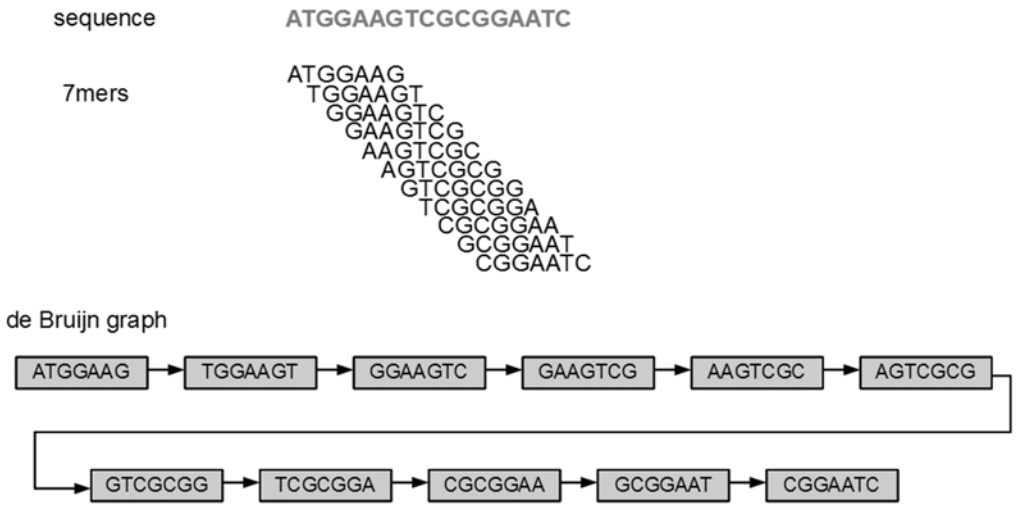

Figura I.10: *(*tomada de `http://www.homolog.us`*) División de una lectura en k-meros y construcción de un grafo de De Bruijn con el conjunto de k-meros para reconstruir la lectura original*

(ii). Construcción de un grafo de De Bruijn con el conjunto de k-meros, de modo que cada nodo contiene un k-mero de longitud k − 1 que se solapa exactamente en k − 2 nucleótidos con otros nodos.

(iii). Búsqueda de los caminos eulerianos que reconstruyen la secuencia original de la que proceden todas las lecturas relacionadas.

Esta estrategia se ve muy afectada por las repeticiones y por los errores de secuenciación que los ensamblajes por solapamiento [63]. Además, como el ADN es de doble cadena, a la hora de reconstruir los caminos eulerianos puede darse el caso que algún k-mero aparezca en el sentido de la transcripción y otro en antisentido, provocando secuencias consenso artefactuales. Algunos ejemplos de los ensambladores más populares que utilizan esta estrategia son Euler-SR [65], Velvet [11], SOAPdenovo [66] y ABySS [67] y el Ray [68] (este último permite la paralelización del ensamblaje).

I.2.6. Estrategias para la secuenciación de transcriptomas.

La secuenciación de pools de ADNc a menudo se utiliza para caracterizar el transcriptoma de un organismo de un modo rápido y barato. El transcriptoma de un organismo engloba al conjunto de genes que se expresan (EST) en una célula o conjunto de células, que incluyen tanto a los ARN que codifican proteínas como los no codificantes (ARNnc). Este tipo de secuenciación proporciona información sobre los genes de un organismo a bajo coste en comparación con la secuenciación de genomas, ya que solo se

investigan las regiones que se transcriben, en lugar del genoma completo. La generación de EST a partir de ARNm se considera la estrategia más frecuente y más útil para descubrir genes [69].

En los casos en los que el análisis transcriptómico está enfocado a estudiar los genes que codifican proteínas, es importante utilizar secuencias de ARN enriquecidas en poli-(A)+, ya que de este modo se reduce en gran parte los ARN no deseados, como los ARN pequeños y los abundantes ARN ribosómicos (ARNr). También es deseable realizar la normalización de las muestras de ARN, ya que así se reduce la aparición de los transcritos más abundantes y se aumenta la de los poco abundantes [70].

I.2.7. Estrategias para la secuenciación de genomas

En la actualidad se utilizan principalmente dos estrategias para abordar la secuenciación de genomas, BAC a BAC (cromosomas artificiales de bacterias), o mediante la secuenciación de todo el genoma (WGS, del inglés whole genome sequencing). La elección de la estrategia más apropiada dependerá del tamaño y de la complejidad del genoma que se quiere secuenciar, así como de las tecnologías utilizadas y del presupuesto disponible. Otra opción más barata y rápida consiste en la secuenciación del transcriptoma para el estudio de los genes que se expresan (EST, etiquetas de secuencias expresadas, del inglés expressed sequence tags) en determinadas condiciones experimentales, útil para obtener información en especies no modelo con genomas complejos.

BAC a BAC: El genoma o un cromosoma que se quiere secuenciar se trocea en fragmentos solapantes de menor tamaño (100 a 200 kpb aproximadamente) que se clonan en vectores BAC (figura I.11-A.1). Después se seleccionan los BAC y se ordenan en un mapa físico (figura I.11-A.2). Al final, cada BAC se secuencia por separado tras una digestión parcial con una enzima de restricción que producirá fragmentos solapantes al azar, en lo que se conoce por el término inglés shotgun (figura I.11-A.3). Al secuenciar los BAC de uno en uno se consigue reducir la complejidad del ensamblaje, de modo que esta estrategia puede resultar útil para genomas con gran cantidad de repeticiones. Sin embargo, crear la genoteca de BAC y mapear cada clon sobre el genoma supone una enorme cantidad de esfuerzo, además de conllevar mucho tiempo y un gran coste económico. Por eso, esta técnica se está utilizando cada vez menos (figura I.12).

WGS: En esta estrategia se evita la generación de genotecas BAC fragmentando directamente el genoma al azar en elementos solapantes de menor tamaño (figura I.11-B.1). Los fragmentos se secuencian directamente con tecnologías de NGS (figura I.11-B.2), o se clonan primero y se secuencian después en el caso de las tecnologías de secuenciación basadas en capilares. Debido a su mayor sencillez, esta es la estrategia de secuenciación de genomas más utilizada hoy en día (figura I.12). A diferencia del caso anterior, aquí toda la complejidad de la secuencia del genoma tendrá que resolverse con aplicaciones bioinformáticas, ya que no se dispone de ninguna información orientadora.

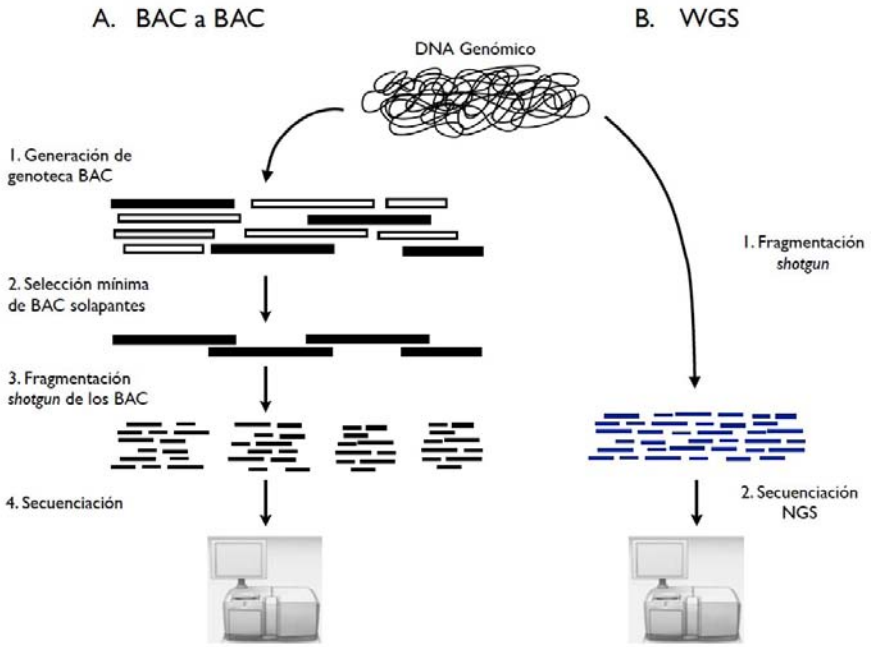

Figura I.11: *Comparación de las estrategias de secuenciación (A) BAC a BAC, y (B) WGS*

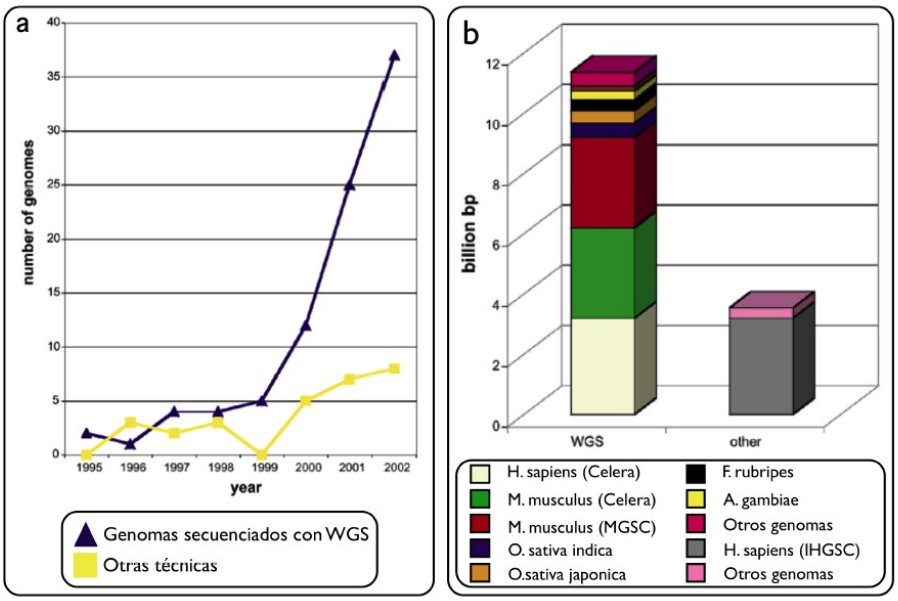

Figura I.12: *(tomada de [71]) Número de genomas secuenciados cada año desde 1995. (a) Número de genomas secuenciados utilizando WGS y con otras estrategias. (b) Cada columna indica el tamaño acumulado (en miles de millones de pares de bases) de los genomas secuenciados por WGS (columna de la izquierda) y por otros métodos (columna de la derecha)*

I.3. Anotación de los ensamblajes

I.3.1. Se anota por similitud

Una vez se han ensamblado los unigenes de un transcriptoma o se han reconstruido las regiones genómicas, se procede a anotar las secuencias ya que el interés de tener las secuencias de un transcriptoma o de un genoma reside en la obtención de información biológica que permita seguir profundizando en el funcionamiento de los organismos y su relación con el entorno. Este incremento de conocimiento científico podrá posteriormente traducirse en aplicaciones prácticas que conlleven beneficios económicos y mejoras en la calidad de vida.

La anotación de genes se lleva a cabo habitualmente de un modo automático e informatizado. Existen muchas herramientas bioinformáticas libres, como Blast [72], que resulta de gran utilidad para encontrar parecidos significativos con genes conocidos que están almacenados en las bases de datos. La anotación por similitud de secuencia tiene algunas limitaciones, en especial en los organismos no modelo, donde el número de genes con anotaciones es menor, sobre todo porque se basan principalmente en la información que se conoce de especies modelo como *Arabidopsis thaliana* [31] y el pez cebra (*Danio rerio*). La anotación basada en similitud será tan exacta como la anotación de las secuencias almacenadas en las bases de datos con las que se compara. Para comparar las secuencias de nucleótidos de los unigenes obtenidos durante el ensamblaje de un transcriptoma suele emplearse con frecuencia Blastx con bases de datos que contienen proteínas conocidas, como RefSeq_protein de GenBank [73] o Swiss-Prot y TrEMBL, ambas de UniProt [74]. Estas dos bases de datos de secuencias de proteínas, tienen diferentes niveles de anotación. Por ejemplo, Swiss-Prot está revisada manualmente, y TrEMBL, utiliza anotaciones automáticas procedentes de referencias cruzadas [74]. Suele ser frecuente, cuando se trabaja con especies no modelo, encontrarse una gran cantidad de ortólogos de función desconocida. Además hay que tener en cuenta que si no se toman las precauciones adecuadas que consisten en la eliminación de los contaminantes en las lecturas originales y de los contigs quimericos, a veces se incorporan a las bases de datos secuencias con errores de anotación o secuencias que realmente son contaminantes o artefactos [44]. Si se considera como buena una secuencia mal anotada, este error también se mantendrá en nuestras secuencias anotadas por similitud. También existe la posibilidad de anotar con Blastn para confirmar que los transcritos reconstruidos con el ensamblaje fueron secuenciados también en otros experimentos de EST.

I.3.2. Anotaciones bioinformáticamente útiles

Gracias a la anotación por similitud se puede enriquecer la información de los transcritos reconstruidos con una definición y con otros elementos de anotación, como la ontología de genes (GO, del inglés Gene Ontology) [75], las rutas metabólicas en las que intervienen los transcritos, recogidas en los mapas KEGG [76], los dominios proteicos registrados en InterPro [77], y en caso de tener actividad enzimática, el código de la EC (del

inglés, Enzyme Commission), además de otras anotaciones no funcionales como SNP, SSR y miRNA.

Gene Ontology: El objetivo del consorcio de la GO es producir un vocabulario controlado y dinámico, aplicable a todos los seres vivos, incluso aunque el conocimiento acerca de la función de los genes y de las proteínas en las células esté en continuo cambio [75]. Con este fin, se desarrollaron tres ontologías independientes: procesos biológicos, funciones moleculares y componentes celulares. La GO, como todas las ontologías, mantiene un lenguaje controlado que es útil para las personas y para los ordenadores, ya que cada elemento de la ontología tiene un código numérico (por ejemplo, GO:1234567) práctico para los análisis bioinformáticos, y una descripción informativa para los usuarios. Además, la GO mantiene una relación jerárquica entre sus elementos. La jerarquización de elementos permite la anotación a distintos niveles, debido al colapsamiento de anotaciones diferentes de la misma rama. Estas ontologías están disponibles en `http://www.geneontology.org/`

Mapas KEGG: Los mapas de rutas metabólicas de KEGG son diagramas gráficos que representan interacciones moleculares y redes metabólicas, procesos con información genética, ambiental, procesos celulares, sistemas orgánicos y enfermedades humanas [76]. Se pueden consultar en `http://www.genome.jp/kegg/`, y son de gran utilidad para relacionar entre sí los genes que participan en una misma ruta, algo muy útil cuando se realizan análisis funcionales.

Código EC: La Comisión Internacional para Enzimas se creó en 1956 en la Unión Internacional de Bioquímica y Biología Molecular para evitar que la misma enzima recibiera diferentes descripciones o nombres. Su función fue aportar un nombre descriptivo sobre la reacción que cataliza la enzima, y un código único para cada función enzimática. Las enzimas se nombran con cuatro números separados por puntos, EC 1.2.3.4, donde el primer número da la característica más genérica y los siguientes son cada vez más específicos. Así, el primer número puede tomar 6 valores; 1 para oxidoreductasas, 2 para las transferasas, 3 para las hidrolasas, 4 para las liasas, 5 para las isomerasas y 6 para las ligasas. Todo lo referente con estos códigos se puede encontrar en `http://www.chem.qmul.ac.uk/iubmb/enzyme/`. Eso sí, solo las proteínas con actividad enzimática tendrán un código EC.

InterPro: La base de datos InterPro integra modelos predictivos de varios repositorios (Pfam, PRINTS, PROSITE, SMART, ProDom, PIRSF, SUPERFAMILY, PANTHER, CATHGene3D, TIGRFAMs y HAMAP). Cada uno se centra en diferentes aspectos biológicos o utiliza una metodología distinta para encontrar el denominador común de las secuencias. El propósito de InterPro es combinar los puntos fuertes de cada uno de los repositorios para poner a disposición de la comunidad científica una única fuente con información estructurada sobre familias de proteínas, dominios y regiones funcionales [77]. InterPro está disponible en `http://www.ebi.ac.uk/interpro/`. Existe una herramienta que se dedica expresamente a encontrar los dominios InterPro para una colección de proteínas concretas: InterProScan (`http://www.ebi.ac.uk/Tools/pfa/iprscan/`).

Ortólogos en especies de referencia: Existen varios organismos modelo cuyos genes han sido identificados y estudiados. Estos genes siendo disponibles en bases de datos como RefSeq (NCBI) [78] o Ensembl [79] con identificadores propios a cada una de éstas. Se utilizan para extrapolar al organismo no modelo los comportamientos que seríamos capaces de detectar si los genes estudiados fueran de la especie modelo.

SNP: Polimorfismo mononucleotídico, del inglés single-nucleotide polymorphism, es una variación en la secuencia de ADN que afecta a una sola base de una secuencia del genoma y se consideran una forma de mutación puntual que ha sido lo suficientemente exitosa evolutivamente para fijarse en una parte significativa de la población de una especie. Este tipo de polimorfismo tiene una gran importancia biológica, ya que determinan la mayor parte de la variabilidad genética de los individuos, causando muchas de las diferencias fenotípicas (observables) de los mismos.

SSR: Las repeticiones de secuencia simple son secuencias de ADN en las que un fragmento se repite de manera consecutiva. Generalmente se encuentran en zonas no codificantes del ADN. Las SSR han sido descritas por primera vez en 1982 [80]. La variabilidad que presentan resulta útil como marcadores moleculares en una gran variedad de aplicaciones en el campo de la genética. Esto se debe a su capacidad para generar una huella genética personal o perfil genético.

MicroARN: es un ARN monocatenario de una longitud de entre 21 y 25 nucleótidos que procede de un precursor que tiene un tamaño entre 75 y 200 pb. Se descubrió recientemente [81] y tiene la capacidad de regular la expresión de otros genes mediante diversos procesos, utilizando para ello la ruta de ribointerferencia.

I.3.3. Programas para anotar

Se han desarrollado muchas herramientas bioinformáticas para obtener información biológica acerca de los productos de los genes que contienen nuestras secuencias. La mayor parte de ellas se basan en la búsqueda de anotaciones por similitud con otras secuencias ortólogas conocidas que están almacenadas en bases de datos. Por ejemplo, la herramienta más conocida y más utilizada para buscar información es Blast [72] que es una herramienta útil para anotación de secuencias, ya que se pueden asignar anotaciones desde los homólogos con mayor similitud. Otra herramienta muy popular y cómoda es Blast2GO [82], que tiene una interfaz fácil de utilizar y ejecutable en cualquier sistema operativo en el que esté instalado Java. Blast2GO ejecuta Blast de modo remoto para determinar a qué se parecen las secuencias que se desean analizar, y de ahí extrae los términos de la Gene Ontology, la nomenclatura de la Enzimme Commission, las rutas KEGG y los códigos InterPro. En 2005 se describió una herramienta de anotación de secuencias transcriptómicas conocida como AutoFact [83]. Este programa realiza búsquedas por similitud utilizando Blast con varias bases de datos (UniRef90, UniRef100, NCBI's nr, COG, KEGG, Pfam, Smart, est_others, LSU Large SubUnit ribosomal RNA, SSU Small SubUnit ribosomal RNA), y entre todas ellas elige la descripción que considera más adecuada, clasificando las anotaciones obtenidas de un

modo estructurado. AutoFact también anota las enzimas con la nomenclatura de la Enzime Commission y las rutas KEGG. Otra herramienta de anotación de secuencias de transcriptómica es Sma3s [84], muy eficaz para anotar grupos grandes de unigenes. A diferencia del Blast2go [82], que normalmente se ejecuta en servidores remotos, el Sma3s puede ser fácilmente instalado y ejecutado en un ordenador local o en un clúster. Además, el hecho de que sea paralelizable reduce drásticamente el tiempo de ejecución y permite tener la anotación de un transcriptoma completo en pocos días. La anotación incluye el nombre del gen, los términos de la Gene Ontology (GO), la nomenclatura de la Enzime Commission (EC) y los códigos InterPro (IPR).

Conviene mencionar que la anotación de secuencias genómicas tiene que seguir un protocolo totalmente diferente, y no vale ninguna de las herramientas anteriores. Una herramienta destacable es Maker [85], creada para la anotación estructural de genomas eucariotas, capaz de identificar repeticiones y alinear EST y secuencias de proteínas a la secuencia genómica de interés para predecir los genes que contiene. También utiliza tres predictores de exones y de uniones entre intrones y exones para poder dar precisión de nucleótido a las predicciones que realiza. El programa fue adaptado posteriormente para los proyectos de ultrasecuenciación [86], de manera que el análisis de los datos se realiza con varias CPU simultáneamente y no le supone ningún problema el análisis de un gran volumen de secuencias.

I.3.4. Bases de datos

Debido a la gran cantidad de datos genómicos, incluidas las secuencias y sus anotaciones, se ha vuelto imprescindible la aplicación de la tecnología informática de bases de datos para mantener y guardar de forma ordenada toda la información biológica que hoy en día se está generando. Y no solo para ofrecerla a la comunidad científica, sino también para poder gestionarla dentro de los laboratorios que las están generando. Estas bases de datos deben tener interfaces fáciles de manejar y herramientas para consultar los datos disponibles, de modo que resulten útiles a usuarios sin una gran experiencia en informática. Por este motivo conviene saber que pueden ser de diferentes tipos en función de cómo se organicen sus datos: jerárquicas, de red, relacionales, orientadas a objetos y documentales. Las bases de datos relacionales son las más utilizadas y las que mejor se adaptan a los proyectos de genómica y transcriptómica.

I.3.4.1. Bases de datos relacionales

Las bases de datos relacionales son el modelo más utilizado en la actualidad para implementar bases de datos ya planificadas (aunque cada vez son más utilizadas las bases de datos tipo Hbase [`https://hbase.apache.org`] o MongoDB [`www.mongodb.com`] para datos genómicos y transcriptómicos). Este tipo de configuración permite establecer interconexiones, relaciones entre los distintos datos almacenados en tablas, de ahí proviene su nombre: "Modelo Relacional". Tienen la ventaja de interrelacionarse sin la necesidad de duplicar una

gran cantidad de información utilizando un lenguaje estándar denominado SQL (Structured Query Language).

El modelo más simple de una base de datos relacional es de tipo sin normalizar. En la figura I.13-A se ve un ejemplo en el cual un unigén tiene una sola tabla Unigene que contiene su nombre (`name`), secuencia (`fasta`), longitud (`length`), la descripción (`description`), y la procedencia (`source`) de la información. El contenido de la descripción puede estar vacío en muchos casos, y también puede que muchos unigenes compartan la misma descripción lo que implica que se estará ocupando en disco un espacio en el que no hay información. De hecho, el diseño del esquema de la base de datos, debería contener una estructura de tablas completamente normalizada, es decir, la información no se almacena de un modo redundante, ya que diferentes tablas de la misma base de datos no almacenan la misma información, y las columnas no almacenan valores repetidos. En el ejemplo de la figura I.13-B, la información sobre la descripción y su procedencia se colocan en una tabla nueva (`Description`) a la que hará referencia un identificador (`description_id`). La principal ventaja de la división en tablas es que los unigenes sin descripción no `ocuparían` registros vacíos en la tabla `Description`, lo que ahorra mucho espacio al no rellenar miles de registros sin información en la tabla.

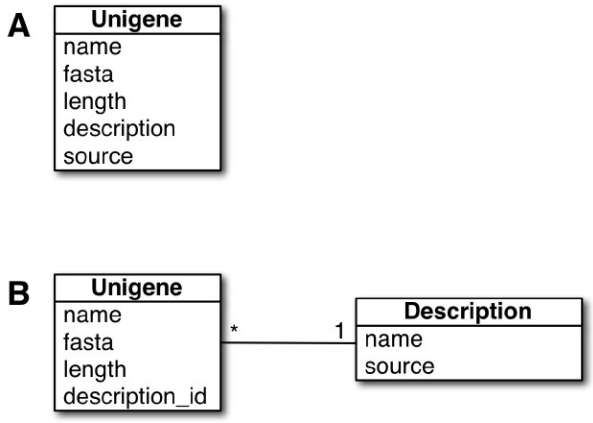

Figura I.13: *Ejemplo de tablas para una base de datos de tipo relacional sin normalizar (A) y normalizada (B)*

En toda base de datos de transcriptómica o genómica, con miles de genes y anotaciones, también resulta esencial contar con un modo de consultar la información fácil y rápido. Para ello se suelen incluir motores de búsqueda de texto que localizan palabras clave dentro de la información de la base de datos, por lo general en las anotaciones de los genes, y muestran al usuario de modo ordenado la información de la base de datos que está relacionada con la búsqueda realizada. Otro modo de buscar información en una base de datos de secuencias consiste en realizar una comparación con Blast para encontrar si la secuencia que se quiere estudiar se parece a alguna de las que hay en la base de datos, y así conocer la información disponible para ella.

Las base de datos utilizadas en este trabajo fueron parte del trabajo de tesis doctoral de Noé Fernández-Pozo [87] y se desarrollaron en Ruby On Rails. Con este entorno de trabajo, el desarrollo de la base de datos se divide en tres partes, modelo, vista y controlador: El modelo indica las relaciones entre las tablas de la base de datos relacional. En la vista se desarrolla la página web que mostrará la información de la base de datos. El controlador es el intermediario que consulta la base de datos relacional y envía los datos necesarios a la vista que se está consultando. El lenguaje más utilizado para consultar y gestionar bases de datos es SQL, aunque con Ruby On Rails se utiliza Ruby y comandos propios de Ruby On Rails para asociar la web con la base de datos y realizar consultas (Ruby On Rails se encarga de traducir estos comandos a SQL para interactuar con la bases de datos, lo que permitiría crear las bases de datos sin conocimientos de SQL). Además, otra ventaja de Ruby On Rails es que el mismo código es válido para interaccionar con bases de datos creadas en SQLite, MySQL y Oracle.

I.3.4.2. Bases de datos internacionales

Los análisis bioinformáticos de las secuencias no serían posible hoy en día si no existieran bases de datos públicas que serán la fuente de información a partir de la cual se puede completar el conocimiento obtenido con los proyectos de secuenciación. Estas bases de datos se pueden dividir en dos grandes grupos: las generalistas y las específicas.

I.3.4.2.1. Generalistas

Estas bases de datos almacenan y ponen a disposición de la comunidad científica muchos tipos de información para un gran número de organismos. En la tabla I.1 están disponibles las direcciones web de las principales bases de datos disponibles en la web, entre las que cabe destacar las siguientes: GenBank [73] (mantenida por el National Center for Biotechnology Information, NCBI, EE.UU.), EMBL [88] (mantenida por el European Bioinformatics Institute, EBI, Unión Europea) y DDBJ [89] (DNA Data Bank of Japan, en Japón). Tienen una colaboración para compartir repositorios [90], de modo que todas las secuencias están disponibles para los usuarios en las tres bases de datos, independientemente de en cual se añadió originalmente. Estas bases de datos está sometida a un crecimiento continuo cada año, junto con la explosión en el crecimiento producida por las NGS [91], lo que está provocando incluso que se anuncie la posibilidad de cerrar algunos repositorios [31], como el de lecturas de NGS, The Sequence Read Archive (SRA) [92] de NCBI. Una alternativa firme para continuar dando soporte a estas lecturas es The European Nucleotide Archive (ENA) [92].

I.3.4.2.2. Específicas

Otro modo muy interesante y útil para abordar el problema de la gran cantidad de datos que se están generando en biología es el desarrollo de nuevas bases de datos

especializadas en sólo un organismo o varios organismos relacionados, o en secuencias que comparten algo en común (tabla I.1). Por ejemplo, está Ensembl [79], que contiene secuencias de genomas completos de vertebrados y otros eucariotas principalmente, o The Zebrafish Model Organism Database (ZFIN) [93] para el pez zebra (*Danio rerio*), que incluye a su genoma y unas herramientas bioinformáticas para consultar su mapa físico y genético, y conocer qué proteínas se codifican en sus genes.

Tabla I.1: *Resumen de bases de datos más representativas y sus direcciones web*

Base de datos	URL
de nucleótidos	
GenBank	http://www.ncbi.nlm.nih.gov/genbank/
EMBL	http://www.ebi.ac.uk/embl/
DDBJ	http://www.ddbj.nig.ac.jp/
de lecturas	
SRA	http://www.ncbi.nlm.nih.gov/sra
ENA	http://www.ebi.ac.uk/ena/
de organismos secuenciados	
Ensembl	http://www.ensembl.org/
Phytozome	http://www.phytozome.net/
PlantGDB	http://www.plantgdb.org/
JGI	http://genome.jgi.doe.gov/
de un organismo modelo	
TAIR	http://www.arabidopsis.org/
FlyBase	http://flybase.org/
MGD	http://www.informatics.jax.org/
ZFIN	http://zfin.org/

I.4. Lenguajes de programación y sistemas operativos

Un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un ordenador debe ejecutar. Por lo tanto, un lenguaje de programación se puede considerar como un modo práctico para dar instrucciones a un ordenador.

Los lenguajes de programación se dividen tres categorías:

- lenguajes interpretados
- lenguajes compilados
- Lenguajes intermediarios

Lenguaje interpretado

Un lenguaje de programación es, por definición, diferente al lenguaje máquina. Por lo tanto, debe traducirse para que el procesador pueda comprenderlo. Un programa escrito en un lenguaje interpretado requiere de un programa auxiliar (el intérprete), que traduce los comandos de los programas según sea necesario.

Lenguaje compilado

Un programa escrito en un lenguaje "compilado" se traduce a través de un programa anexo llamado compilador que, a su vez, crea un nuevo archivo independiente que no necesita ningún otro programa para ejecutarse a sí mismo. Este archivo se llama ejecutable.

Lenguajes intermediarios

Algunos lenguajes pertenecen a ambas categorías dado que el programa escrito en estos lenguajes puede, en ciertos casos, sufrir una fase de compilación intermediaria, en un archivo escrito en un lenguaje ininteligible (por lo tanto diferente al archivo fuente) y no ejecutable (requeriría un intérprete).

Los lenguajes que se han manejado en este trabajo han sido los siguientes:

Perl: es un leguaje de scripting orientado a objetos ampliamente utilizado y de los más rápidos dentro de los lenguajes interpretados. Se utilizó para el análisis automático de los resultados de programas de ensamblaje, para crear script de generación de lecturas artificiales y para desarrollar el programa de ordenación y orientación de contigs, ICMapper. La página de referencia de este lenguaje es `http://www.perl.org/`

Ruby: es un leguaje de scripting interpretado orientado a objetos, capaz de crear aplicaciones complejas de un modo rápido, manteniendo el código legible y corto [94]. Se utilizó para el desarrollo del programa Cominer. También se utilizó para crear múltiples scripts para manipular secuencias, anotaciones y otros ficheros con grandes cantidades de datos. La última versión utilizada al término de este trabajo fue la 1.9.3. La página de referencia de este lenguaje es `http://www.ruby-lang.org/es/`

Ruby On Rails (ROR): es un entorno de desarrollo de páginas web con base de datos que fue utilizado anteriormente [87] para desarrollar base de datos de transcriptómica. En el presente trabajo se utilizó la misma implementación de base de datos para SoleaDB, IsochrysisDb y RuditapesDB y fue necesario manejar el entorno Ruby on Rails para adaptar estas bases de datos a las necesidades del momento e introducir cambios. La última versión utilizada al término de este trabajo fue la 2.3.11. La página de referencia de este lenguaje es `http://www.rubyonrails.org.es/`

HTML: es el lenguaje utilizado para el desarrollo de páginas web. Se utilizó en las bases de datos SoleaDB.

Sistemas operativos:

UNIX OSX 10.6: Sus comandos se utilizaron para la ejecución de programas y para la visualización y manipulación de datos en general.

UNIX SLES 10.2.5: Suse Linux Entrerprise Server se utilizó para la ejecución de programas en el sistema de colas de los supercomputadores Picasso SuperDome y Picasso-cluster, y para la consulta y manipulación de datos en general.

I.5. Interés económico de las especies marinas

Los recursos marinos son vitales dada la creciente demanda mundial de proteína animal, derivada del aumento demográfico (8 mil millones de seres humanos para 2030) y de los hábitos dietéticos actuales, que relacionan la salud y el consumo de pescado [95, 96]. El pescado es considerado un alimento fundamental en la dieta, ya que aporta proteínas de fácil digestión, ácidos grasos esenciales de la serie n-3 e importantes minerales y vitaminas, sobre todo A y D [97]. Su consumo ayuda a un adecuado equilibrio de ácidos grasos poliinsaturados n-3/n-6, relacionado con la expresión génica, lo que previene de modo natural algunas de las actuales enfermedades crónicas o degenerativas [98].

La producción pesquera mundial ha aumentado de forma constante en las últimas cinco décadas (figura I.14). El suministro de peces comestibles se ha incrementado a una tasa media anual del 3,2 %, superando así la tasa de crecimiento de la población mundial del 1,6 %. El consumo aparente mundial de pescado per cápita aumentó de un promedio de 9,9 kg en la década de 1960 a 19,2 kg en 2012, según las estimaciones preliminares [99]. Este incremento notable se ha debido a una combinación de crecimiento demográfico, aumento de los ingresos y urbanización, y se ha visto propiciado por la fuerte expansión de la producción pesquera y la mayor eficacia de los canales de distribución.

Una gran parte de los recursos marinos son obtenidos a partir de productos de la pesca extractiva (figura I.14). Mientras la producción mundial de este tipo de pesca conoce una estabilidad en las últimas décadas, la producción de la acuicultura sigue creciendo y está alcanzando mundialmente un desarrollo espectacular (figura I.14), lo que lo convierte en el sector alimentario de más rápido crecimiento. Esta tendencia tiene que mantenerse o subir, para sustentar los actuales niveles de consumo de productos pesqueros, pues la población mundial sigue su aumento geométrico.

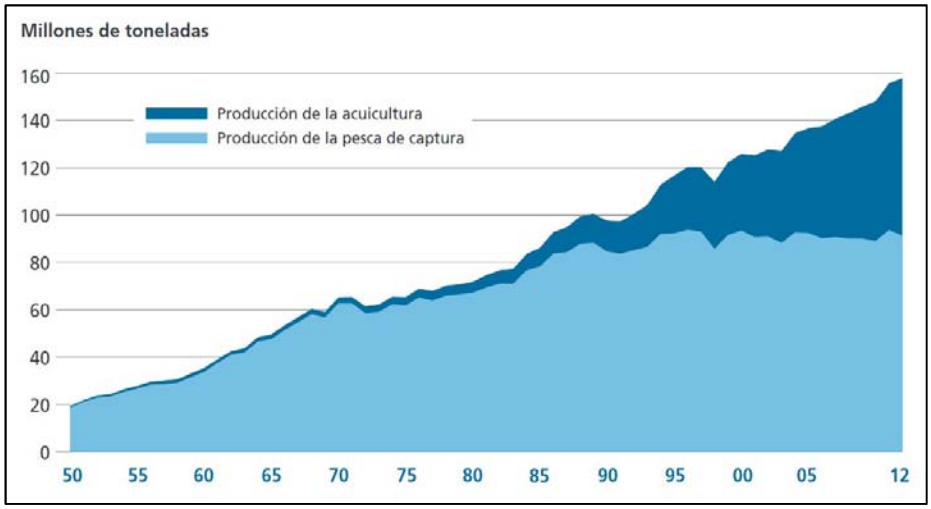

Figura I.14: *(tomada de [99]) Producción mundial de la pesca de captura y la acuicultura*

Aunque sea de relativamente reciente aparición, la acuicultura se ha caracterizado por demandar desde el principio una diversificación de especies que se puedan producir industrialmente. Fruto de esta precocidad, se emprendieron de forma generalizada numerosas actuaciones dirigidas a conocer las posibilidades de cultivo de un notablemente elevado número de peces. Durante los últimos tres lustros, son más de treinta las especies de peces que han sido objetivo de estudio en el área mediterránea [100]. Sin embargo, este esfuerzo se ha visto poco reflejado en la producción acuícola, que continúa monopolizada por las especies inicialmente desarrolladas, es decir: dorada (*Sparus aurata*) y lubina (*Dicentrarchus labrax*). Las razones para este desfase se podrían resumir en la dificultad tecnológica y la inadecuada posición de mercado de las numerosas especies exploradas.

En España, estas dos especies junto con los mejillones forman las especies de acuicultura pioneras y ampliamente comercializadas. En la figura I.15 se puede ver el valor de la producción de éstas en comparación con otras menos comercializadas. Sin embargo, la saturación alcanzada en el mercado por su producción ha traído consigo la diversificación de especies acuícolas como el cultivo del besugo (*Pagellus bogaraveo*), el pargo (*Pagrus pagrus*), el abadejo (*Pollachius pollachius*), la corvina (*Argyrosomus regius*), el atún (*Thunnus thynus*) y el lenguado senegalés (*Solea senegalensis*, Kaup 1858). Éste, junto con el lenguado común (*Solea solea*), desde los últimos treinta años aparecen en el sur de Europa como unos buenos candidatos para la diversificación de los mercados europeos. Una vez superados los problemas iniciales, el lenguado ha demostrado ser una prometedora especie para la acuicultura marina [101]. Existen estudios relacionados con su acuicultura en el litoral gaditano y en estuarios portugueses de hace ya más de veinte años [102]. Uno de los principales problemas con los que se encontró el cultivo del lenguado en sus inicios fueron las bajas tasas de crecimiento en juveniles y la aparente gran susceptibilidad a enfermedades, sobre todo pasteurelosis, vibriosis, mixobacteriosis y enfermedades virales.

Estos problemas han sido en parte solventados a lo largo del tiempo mediante la mejora de las dietas y el estudio del estrés debido a las grandes densidades de cultivo. Sin embargo

existen otros problemas relacionados con la reproducción en cautividad, el cultivo larvario, la nutrición y la respuesta immunitaria. Las cuales podrían ser solventados más fácilmente mediante un conocimiento en profundidad de los mecanismos moleculares y fisiológicos implicados.

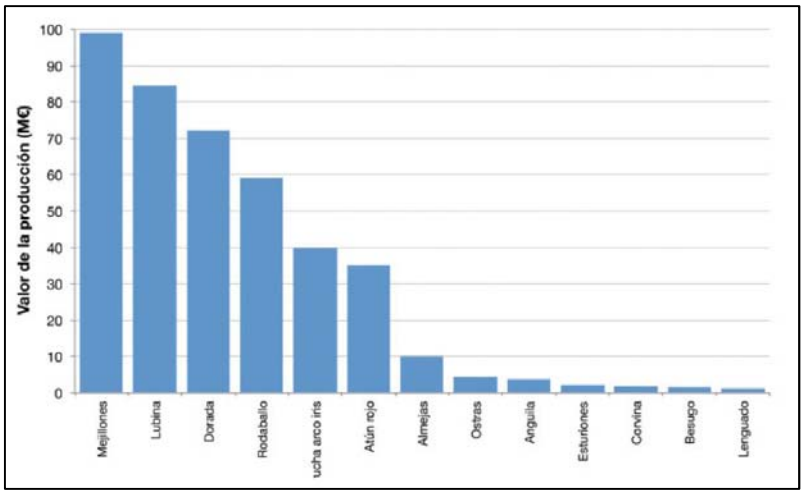

Figura I.15: (tomada de [103]) Valor de la producción (primera venta) de las especies de acuicultura marina (por grupos) en España en 2012, en millones de euros (JACUMAR)

Parte II
Objetivos

II. Objetivos

El objetivo general de esta tesis es de mejorar el conocimiento a nivel genómico del lenguado senegalés (*Solea senegalensis*), el lenguado común (*Solea solea*), y de algunas especies afines a través de un análisis bioinformático y del desarrollo de protocolos de almacenamiento de los datos genómicos de estas especies, por el fin de suministrar elementos útiles que puedan abrir vías en la mejora genética de las mismas. Para ello se seleccionarán las mejores herramientas bioinformáticas disponibles y se optimizará su uso para sacar un rendimiento óptimo de los datos brutos disponibles. Así mismo, se desarrollarán aquellas nuevas herramientas y flujos de trabajo que sean necesarias para facilitar el tratamiento de los datos y mejorar la calidad de los resultados. Tanto los datos originales como los procesados se integrarán finalmente en una base de datos para ofrecer a la comunidad científica un transcriptoma y genoma depurado con las anotaciones más útiles.

Los objetivos generales presentados anteriormente pueden desglosarse en los siguientes subobjetivos más específicos:

1- Selección y comparación de los programas de ensamblaje de las secuencias disponibles de una parte los programas específicos de lecturas largas, de otra parte aquellos específicos de lecturas cortas. Además, establecer un modelo para el testeo, selección y optimización de los programas de ensamblaje que se van a utilizar y métodos para su uso combinado.

2- Desarrollo de flujos de trabajo para el ensamblaje del transcriptoma y del genoma de la especies problema de este trabajo. Estos flujos incluyen tanto las herramientas disponibles en la bibliografía como las herramientas propias con el fin de utilizar los dos tipos de datos (lecturas largas o cortas) o una combinación de ambos, y proporcionar el resultado más optimizado para la posterior anotación.

3- Inclusión la información en una base de datos web previamente desarrollada que permita almacenar los datos de modo estructurado y extensible para facilitar el acceso a ellos por parte de la comunidad científica. Aportar unas mejoras a la base de datos tanto en los tipos de información disponible como en la interfaz, jerarquía y organización de los datos así que la forma de acceder a ellos.

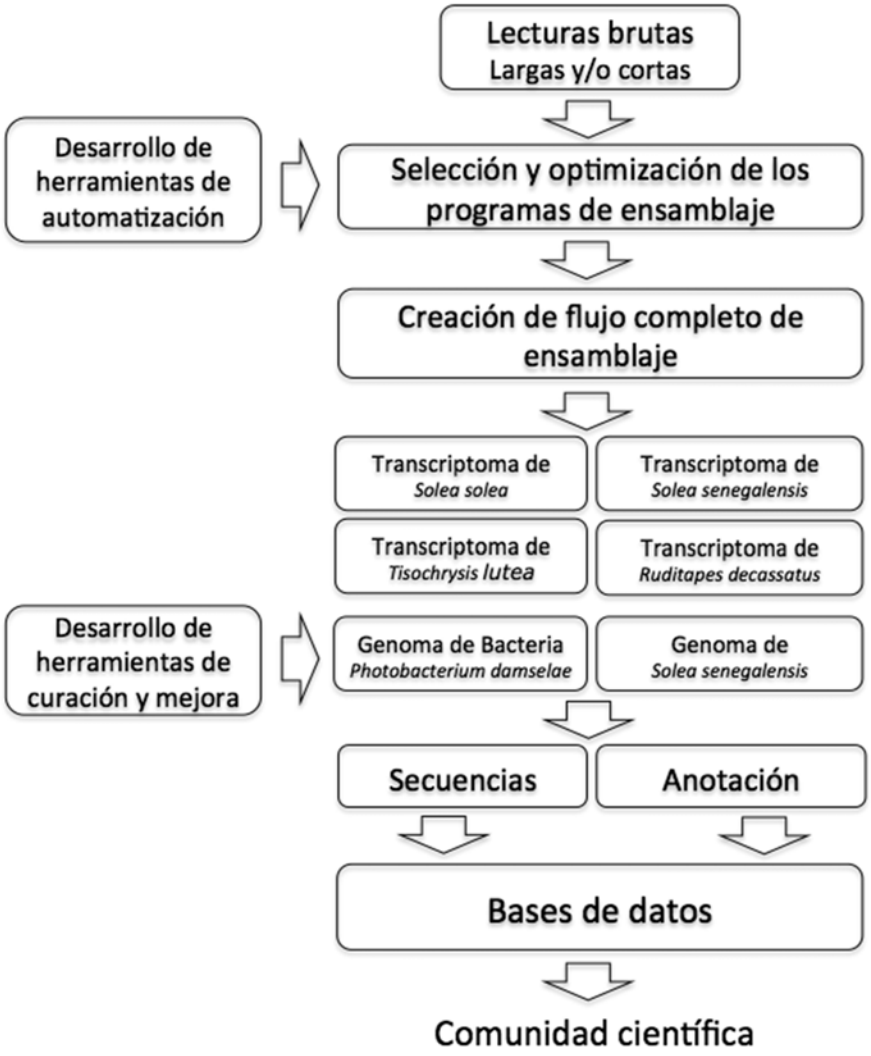

Figura II.1: *Esquema de los objetivos. Creación de flujo completo de ensamblaje de las lecturas de transcriptomica. Elaboración de herramientas para el testeo y selección previa de los ensambladores usados y otras para la post evaluación y la mejora de los resultados. Posteriormente la información se integra en bases de datos disponibles para la comunidad científica*

Parte III
Materiales y métodos

III. Materiales y métodos

III.1. Equipos informáticos

Durante el desarrollo de esta tesis se han utilizado varios ordenadores personales y supercomputadores, todos ellos con sistema UNIX. Los supercomputadores forman parte de la infraestructura del Centro de Supercomputación y Bioinformática que la Universidad de Málaga tiene en el edificio de Bioinnovación del Parque Tecnológico de Andalucía.

Hasta 2012, el trabajo de esta tesis se realizó en dos supercumputadores: Picasso SuperDome y Picasso-Cluster.

Picasso-cluster: Se trata de un cluster de 80 núcleos Intel Xeon E5450 a 3,00 GHz (con arquitectura x86) y con 160 GB de RAM, divididos en 10 blades con 8 núcleos y 2 GB de RAM para cada uno. Los blades están interconectados por red InfiniBand. El gestor de colas es PBS Pro y el sistema operativo es el Suse Linux Entrerprise Server v. 10.2.5.

Picasso: se trata de un HP SuperDome con 128 núcleos Intel Itanium2 a 1,6 GHz y con 400 GB de memoria compartida. Ordenador y memoria se encuentran en dos racks conectados entre sí. Un tercer rack contiene el sistema operativo y el scratch (espacio del disco duro para el almacenamiento de datos temporales). El gestor de colas es PBS Pro y el sistema operativo es el Suse Linux Entrerprise Server v. 10.2.5.

A principios del año 2013, se actualizaron todos los recursos de supercomputación, y todas las maquinas comparten ahora la misma arquitectura y están unificados con un único sistema de cola que envía los trabajos a los nodos más adecuados. Las nuevas instalaciones se componen en varios tipos de máquinas:

- Clúster Intel E5-2670 con 768 cores a 2.60 GHz por core y con un RAM total de 3 TB y dispone de una red infiniband.
- Máquinas de memoria compartida con 2 TB de RAM cada una. Tiene 560 cores a 2.40 GHz por core. Dispone de una red infiniband.
- Clúster AMD Opteron 6176 con 984 cores, 4 TB de RAM y 246 TB de scratch.
- Clúster de visualización ESX para máquinas virtuales :
    - 2 máquinas a 2.66 GHz y 32 GB de RAM cada una.
    - 3 máquinas a 2.66 GHz y 16 GB de RAM cada una.
    - 4 máquinas a 2.13 GHz y 124 GB de RAM cada una.
    - Almacenamiento compartido con un total de 750 TB (brutos).

III.3. Programas informáticos

III.3.1. De uso general

III.3.1.1. SLURM 2.4.4:

Slurm es un gestor de recursos de código abierto diseñado para clusters de computación de cualquier tamaño con sistema operativo Linux. Entre los recursos gestionados por Slurm se incluyen nodos, particiones que agrupan nodos, ejecuciones o trabajos, número de CPU, cantidad de memoria, tiempo y otros recursos compartidos como las GPU (tarjetas gráficas).

Las particiones se pueden considerar como colas de trabajos, cada una de ellas con un conjunto de restricciones relativas al tiempo de ejecución y al número de trabajos que pueden ser enviados a esa cola por parte del usuario.

Cuando el usuario desea realizar una ejecución, específica la cantidad y tipo de los recursos que va a consumir, y en base a estas especificaciones y a las ejecuciones ya solicitadas por el propio usuario, SLURM le asigna una prioridad al trabajo. Va asignando trabajos por orden de prioridad hasta que los recursos disponibles se agotan. Los trabajos para los cuales no hay recursos en ese momento se quedan en espera y serán evaluados de nuevo cuando los recursos que necesita sean liberados.

Para enviar un trabajo al sistema de cola, se crea un fichero bash, que llamaremos my_job.sh, con la siguiente estructura:

```
##numero de cpus a reservar para el trabajo
#SBATCH --cpus=4
##cantidad de memoria a reservar para el trabajo
#SBATCH --mem=4gb
##tiempo maximo asignado al proceso
#SBATCH --time=4-00:00:00
##fichero log de salida
#SBATCH --error=job.%J.err
## fichero donde se muestran los errores
#SBATCH --output=job.%J.out

## linea de comando para ejecutar el programa
my_program -input [options]
```

III.3.1.2. Array-Jobs

No se trata realmente de un programa, sino de cómo aprovecharse de la ejecución del sistema de colas para ejecutar en paralelo (nunca de forma distribuida) algunos programas que no están pensados en paralelo, siempre que el conjunto de datos de entrada se pueda dividir en conjuntos más pequeños que se pueden analizar de forma independiente. Supongamos que la entrada es un fichero en formato fasta y que el programa analiza cada secuencia una a una de modo independiente (por ejemplo, sería el caso de Blast o de los programas de preprocesamiento y anotación, pero no sería aplicable a programas de

ensamblaje). Con el uso de los array-jobs se dividirá el fichero de entrada en varios «subficheros» con el mismo formato y un número menor de secuencias, y cada uno de estos «subficheros» se ejecutará independientemente en paralelo. Para dividir el fasta original se puede utilizar un script Ruby simple como split_fasta.rb [87], al que solo hay que pasar dicho fichero original que contiene todas las secuencias y el número de secuencias que va a tener cada uno de los «subficheros» que se crearán, tal como se indica en la siguiente línea de comandos:

```
ruby split_fasta.rb my_file.fasta 500
```

Con el que se crearán «subficheros» de 500 secuencias cada uno.

Para ejecutar el programa en paralelo con cada uno de los «subficheros» se crea un fichero de bash, que llamaremos array_job.sh, con la siguiente estructura:

```
##nombre del trabajo
#SBATCH --job-name=nombre
##numero de cpus
#SBATCH --cpus=4
##cantidad de memoria
#SBATCH --mem=4gb
##tiempo limite
#SBATCH --time=4-00:00:00
##ficheros log
#SBATCH --error=job.%J.err
#SBATCH --output=job.%J.out

## rango de valores para los trabajos
#SARRAY --range=1-20
##creamos una carpeta con un índice para cada trabajo
mkdir dir_${SLURM_ARRAYID}
##movemos el fasta a la carpeta nueva
mv my_file_part${SLURM_ARRAYID}.fasta dir_${SLURM_ARRAYID}

## nos situamos dentro de la carpeta nueva
cd dir_${SLURM_ARRAYID}

## ejecutamos el programa
my_program -input [options]
```

En el fichero del ejemplo se indica que cada uno de los paquetes de secuencias se ejecuta en paralelo utilizando 4 núcleos por cada trabajo y una memoria total de 4 Gb de RAM. En caso de que el programa no pueda ejecutarse en paralelo debe utilizarse una CPU únicamente. El número de CPU es conveniente que sea múltiple del número total de CPU que tiene el supercomputador en el que se ejecuta. Por ejemplo, en los nodos de cálculo de Picasso hay 16 CPU por cada blade, lo que limita la ejecución a un máximo de 16 CPU por paquete, así que utilizando 4 CPU podrán ejecutarse 4 paquetes en cada blade. Sin embargo en los nodos "bigmem" donde hay 64 CPU por cada blade, se pueden ejecutar 16 paquetes por cada blade.

Los programas que no se ejecutan en paralelo pueden aprovechar mejor las CPU disponibles en caso que se indique la opción ntasks al lugar de ncpus:

```
#SBATCH --ntasks=20
```

en este caso el sistema de colas no se esforzaría a encontrar nodos enteros libres para mandarles los trabajos, si no que afectará los paquetes a las CPUs que estén libres en cualquier nodo. Cuando se utiliza la opción 'ntasks', la cantidad de memoria indicada en `#SBATCH --mem` será la memoria consumida por cada trabajo en cada CPU.

En las siguientes líneas del fichero se indica el número de paquetes que contiene el array-job (#SARRAY --range=1-20), que en nuestro caso será 20, de modo que la variable SLURM_ARRAYID irá cambiando su valor desde 1 hasta 20. Después se crea una carpeta para cada uno de los paquetes, y se ejecuta el programa dentro de esa carpeta. Es importante ejecutar cada paquete en una carpeta diferente, porque algunos programas crean archivos temporales que al coincidir en un solo directorio, pueden provocar que los más recientes se sobrescriban sobre los más antiguos, perdiendo información imprescindible y provocando errores graves. Una vez configurado a las necesidades, el fichero .sh se envía al sistema de colas con la orden:

```
sarray array_job.sh
```

con lo que todos los paquetes de secuencias se envían al sistema de colas para que se ejecuten simultáneamente, tantos a la vez como permita el sistema de colas.

III.3.1.3. Gemas de Ruby

Se trata de paquetes autónomos para el lenguaje de programación Ruby con un formato estándar que proporcionan una funcionalidad estándar (serían el equivalente de los plugins en otros programas). Se puede encontrar información detallada de estas gemas en `http://rubygems.org/`. En la misma web se puede encontrar las gemas que fueron desarrolladas en el Centro de Supercomputación y Bioinformática de la UMA (SCBI). Todas se instalan con el comando:

```
gem install nombre_de_la_gema
```

Las gemas de Ruby y ROR utilizadas en algún momento en este trabajo son:

- **bio (v 1.4.2):** gema necesaria para utilizar Bioruby [104] fue de utilidad para interactuar con las variables disponibles en el interfaz de programación de aplicaciones (Application Programming Interface, API) de la web de KEGG pathway. Este método se utilizó para resaltar las enzimas contenidas en las rutas metabólicas de las secuencias de las bases de datos construidas en este trabajo. Enviando la información a través del API de KEGG se indica que enzimas de la

ruta deben tener el fondo de color amarillo. Esta gema se estuvo usando hasta 2013 debido a un cambio que hubo en el API de KEGG.

- **JSON (v 1.4.6):** la usamos en una de las versiones del programa CoMiner ya que es útil para generar y leer ficheros en formato JSON y pasar los objetos a memoria de un modo ordenado.

- **mysql (v 2.7):** nos permitió utilizar mySQL en las bases de datos de ROR.

- **rails (v 2.3.5):** nos permitió instalar ROR.

- **sqlite3-ruby (v 1.2.4):** nos fue útil para utilizar SQLite3 en las bases de datos de ROR.

- **will_paginate (v 2.3.15):** nos sirvió para paginar en las vistas de ROR.

- **xml-simple (v 1.0.12):** la usamos para procesar ficheros en formato xml y pasarlos a objetos en memoria de un modo ordenado.

Las gemas utilizadas en este trabajo se diseñaron específicamente para alguno de los programas que aparecen en este manuscrito, en todos los casos gracias a la colaboración de otros miembros del grupo de investigación, cuyo nombre se especifica en cada una:

- **full_lengther_next (v 0.2.1):** la usamos para instalar Full-LengtherNext [46, 87].

- **scbi_ace (v 0.0.6):** nos permitió indexar los nombres de las secuencias de un fichero en formato ACE y para acceder a todos los datos de un fichero .ace en objetos de Ruby [46].

- **scbi_fasta (v 0.1.9):** la usamos para leer ficheros fasta, con o sin calidades y pasar nombre y secuencia a objetos de ruby [46].

- **scbi_fastq (v 0.0.18):** la usamos para leer ficheros fastq y pasar nombre y secuencia a objetos de ruby [46].

- **scbi_blast (v 0.0.43):** la usamos para ejecutar el Blast y pasar los elementos del alineamiento a objetos de ruby [46].

- **seqtrimnext (v 2.0.59):** la usamos para instalar SeqTrimNext [46].

III.3.1.4. Aptana Studio

Aptana Studio es un entorno de desarrollo integrado de software libre basado en eclipse y desarrollado por Aptana, Inc., que puede funcionar bajo Windows, Mac y Linux y permite la elaboración de aplicaciones web dinámicas que empleen PHP, Ruby, Ruby on Rails y Python además que tiene la posibilidad de incluir complementos para nuevos lenguajes y funcionalidades.

Aptana Studio se ha utilizado para gestionar las webs de bases de datos, además de la realización de todos los scripts y programas que se desarrollaron en este marco de este trabajo. Se puede descargar de `http://www.aptana.com`.

III.3.2. Para el ensamblaje de secuencias

III.3.2.1. CAP3

Programa de ensamblaje de secuencias de tipo OLC, basado en el solapamiento entre los extremos de las secuencias [105]. En los estudios de transcriptómica se considera como uno de los ensambladores más fiables para secuencias de tipo Sanger [106] y también para reensamblar varios ensamblajes procedentes de un mismo conjunto de secuencias para conseguir un resultado que se ajuste más a la realidad [107, 108]. Se puede descargar en `http://seq.cs.iastate.edu/cap3.html`, o ejecutar on-line en la web del SCBI-PAB, `http://www.scbi.uma.es/cap3`. Este programa permite ajustar numerosos parámetros que en la versión web se encuentran rellenos con los valores por omisión para ensamblar EST; si se fueran a ensamblar secuencias genómicas o reensamblar unigenes de transcriptoma habría que incrementar el valor de identidad al 90 %. Por línea de comandos sólo hay que indicarle el fichero fasta y los parámetros que varíen entre los valores por omisión:

```
cap3 my_file.fasta -p identity_value
```

CAP3 proporciona varios ficheros de salida: el fichero con extensión .ace que contiene la información del ensamblaje, un fichero con extensión .contig, donde se encuentra la secuencia consenso de los contigs formados en formato fasta, y otro fichero con extensión .singlets, donde se incluyen las secuencias de los singulones en formato fasta. En caso de redirigir la salida en pantalla a un fichero, se obtendrían los alineamientos del ensamblaje en un fichero de texto.

Este programa al ser compatible con lecturas largas, lo utilizamos en los ensamblajes de transcriptomica para combinar los contigs de otros ensambladores con los parámetros por defecto que son optimizados para ensamblar las lecturas largas.

III.3.2.2. Mira

Se trata de un programa de ensamblaje de tipo OLC, es decir, basado en el solapamiento de los extremos de las secuencias, como CAP3.

Es la evolución de miraEST [109] para adaptarse a secuencias de Roche-454 y otras secuencias de NGS, según se indica en su página web oficial (http://sourceforge.net/apps/mediawiki/mira-assembler/index.php). Se puede descargar de http://sourceforge.net/projects/mira-assembler/files/ y son muchos los parámetros que permite ajustar para cada ejecución; por eso dispone de una buena documentación (http://mira-assembler.sourceforge.net/docs/DefinitiveGuideToMIRA.pdf). Además, hay disponible una versión web del programa en el SCBI-PAB (http://www.scbi.uma.es/mira), y también se puede ejecutar por línea de comandos utilizando el sistema de colas de Picasso (véase el apartado III.3.1.1). A continuación se muestra un ejemplo del fichero de bash necesario para la ejecución de MIRA3 en el sistema de colas. En este ejemplo se combinan secuencias de tipo Sanger y de 454 para ser ensambladas juntas:

```
##numero de cpus
#SBATCH --cpus=16
##cantidad de memoria
#SBATCH --mem=200gb
##tiempo limite
#SBATCH --time=7-00:00:00
##ficheros log
#SBATCH --error=job.%J.err
#SBATCH --output=job.%J.out

# se inicializa el mira en picasso
module load mira

#se crean variables
#nombre del proyecto
export NAME=my_project_name
#carpeta en scratch donde se guardaran los archivos temporales
export SCRATCH=/mnt/scratch/users/cvi_114_uma/hicham/temp/$NAME
mkdir -p $SCRATCH


## ejecutamos el mira
mira -fastq -project=my_project_name --job=denovo,est,normal,454
-CL:ascdc 454_SETTINGS -CO:fnicpst=yes -notraceinfo
COMMON_SETTINGS -GE:not=16 -DI:lrt=$SCRATCH > output.txt
```

MIRA3 solo fue posible ejecutarlo en nodos Bigmem porque son los únicos con memoria compartida y, por lo tanto, capaces de satisfacer los requisitos de RAM para su ejecución. Los ficheros de entrada deben nombrarse como my_project_in.454.fasta. A continuación se describen los parámetros utilizados para realizar el ensamblaje mixto con secuencias de tipo 454 del ejemplo anterior:

Para más información consúltese el manual del programa, mencionado al comienzo de este apartado.

MIRA3 fue uno de los programas que utilizamos para ensamblar las lecturas largas de los transcriptomas tratados en este trabajo.

III.3.2.3. EULER-SR

EULER [110] es un algoritmo para ensamblaje de secuencias cortas y de los primeros que utiliza un camino euleriano implementado mediante un grafo de Bruijn. Según sus autores tiene la ventaja con respecto a los mecanismos de diseño por solapamiento de que es mucho más eficaz con las repeticiones ya que en vez de enmascararlas genera un grafo que permite crear una estructura de repeticiones del genoma para poder tratarlas. El software EULER fue desarrollado para lecturas Sanger y fue posteriormente modificado y denominado EULER-SR [111, 112] para lecturas cortas de 454 GS20 [9], lecturas cortas no pareadas de Illumina/Solexa [113].

EULER se basa en tres principios fundamentales:

- Representa las lecturas como enlaces y los solapamientos como nodos en un grafo de Bruijn.
- Realiza un ensamblaje eficiente mediante la reducción a un problema de camino euleriano: cada enlace debe visitarse una sola vez.
- Las repeticiones se tratan mediante el uso de múltiples enlaces para una lectura repetida.

Es fácil de ejecutar, basta con indicar el fichero de entrada y el valor de k-mero que se quiere utilizar para obtener el resultado en el fichero my_results_file.txt:

```
Assemble.pl my_file.fasta kmer_value > my_results_file.txt
```

Para las secuencias de transcriptómica 454, Euler fue el ensamblador de tipo De Bruijn que mejores resultados devolvía, por lo que lo utilizamos como parte del flujo de trabajo del ensamblaje de este tipo de lecturas.

III.3.2.4. Velvet

Velvet [11, 114] es un conjunto de algoritmos desarrollados por Daniel R. Zerbino y Ewan Birney para el tratamiento de grafos de de Bruijn para ensamblaje de secuencias genómicas.

Velvet hace un uso bastante extenso de herramientas de simplificación de grafos para reducir caminos no intersectantes con nodos simples. Esta simplificación comprime el grafo sin pérdida de información. El programa ejecuta la fase de simplificación durante la

construcción del grafo y, de nuevo, en varias ocasiones durante el proceso de ensamblaje. Esta técnica, introducida como «eliminación de únicos» para grafos de k-mero es similar a la formación de unitigs en los grafos de solapamiento [61] y en los ensambladores OLC [61].

La ejecución de Velvet se hace en dos etapas de la forma siguiente:

```
srun velveth out_folder my_kmer -fastq -longPaired -separate long_paired_reads1.fastq
long_paired_reads2.fastq -long long_single_reads.fastq -shortPaired -separate
short_paired_reads1.fastq short_paired_reads2.fastq > salida1

srun velvetg Velveth_29 -exp_cov auto -cov_cutoff 10 > salida2
```

En la primera línea de comando, `my_kmer` se reemplaza con al valor del k-mero, en la última versión de Velvet este valor puede ir hasta 101. `-fastq` es para indicar que los ficheros introducidos son de tipo .fastq. `-longPaired` indica que las lecturas son largas y pareadas, `-separate` indica que las parejas de lecturas vienen en dos ficheros independientes, `-long` indica que las lecturas son largas y single, `-shortPaired` indica que las lecturas son cortas y pareadas y por ultimo `out_folder` es el nombre de carpeta donde se guardarán los resultados de la ejecución.

En la segunda línea de comando, `out_folder` representa la carpeta creada anteriormente, `-exp_cov` indica la cobertura esperada y `-cov_cutoff` indica la cobertura mínima.

Velvet formó parte del estudio que hicimos sobre la comparación de los programas de ensamblaje para las lecturas largas transcriptomicas y genómicas.

III.3.2.5. Oases

Oases [115] es un extensión de Velvet desarrollada por Marcel Shulz y Daniel Zerbino, destinada a ensamblar lecturas de transcriptómica . Oases importa un ensamblaje preliminar producido por Velvet [11, 114], y agrupa los contigs en grupos pequeños llamados loci. Luego aprovecha de las lecturas pareadas y las lecturas largas, para construir isoformas utilizando la similitud entre los grafos de Bruijn y los grafos de ayuste (splicing).

La ejecución de Oases se compone de tres líneas de comando. Las dos primeras corresponden a una ejecución normal del Velvet tal como se describió más arriba con la diferencia de que en el segundo comando se añade la opción -read_trkg yes que permite el seguimiento de las posiciones de lecturas en el ensamblaje. En tercera línea es donde interviene el Oases recuperando los datos preliminares de Velvet contenidos en `out_folder`.

```
srun velveth out_folder my_kmer -fastq -longPaired -separate long_paired_reads1.fastq
long_paired_reads2.fastq -long long_single_reads.fastq -shortPaired -separate
short_paired_reads1.fastq short_paired_reads2.fastq > salida1

srun velvetg out_folder -read_trkg yes > salida2

srun oases out_folder -ins_length 169 -ins_length_sd 47 -min_trans_lgth
100 -cov_cutoff 3 -min_pair_count 20 > salida3
```

En la tercera línea donde interviene el programa Oases, la opción `-min_pair_count` indica el número de pares de lecturas necesarios para considerar una conexión entre dos contigs montar un scaffold.

Oases se utilizó en el ensamblaje de las lecturas de *Solea senegalensis* y *Solea solea* con los parámetros que dieron el mejor resultado en la fase de pruebas.

III.3.2.6. SOAPdenovo-trans

Es un ensamblador *de novo* de transcriptomas basado en el SOAPdenovo2 [116] y utiliza un algoritmo de ensamblaje paralelizable basado en grafos De Bruijn. Está preparado para manejar el ayuste alternativo y a los diferentes niveles de expresión en los transcritos.

El paquete el programa consiste en dos ejecutables cuyos la utilización de uno u otro depende del k-mero usado.

- SOAPdenovo-Trans-31kmer: para k-meros hasta 31
- SOAPdenovo-Trans-127mer: para k-meros hasta 127

La ejecución del programa se efectúa en cuatro etapas de la forma siguiente:

```
SOAPdenovo-Trans-31kmer pregraph -s config_file -K 29 -p 16 -d 1 -o my_assembly
SOAPdenovo-Trans-31kmer contig -g my_assembly
SOAPdenovo-Trans-31kmer map -s config_file -g my_assembly -K 29 -p 16
SOAPdenovo-Trans-31kmer scaff -g my_assembly -F -p 16
```

- En la primera línea donde se utiliza el comando `pregraph`, `-s` indica el fichero de configuración donde viene la información relativa a las lecturas que se van a utilizar (ver descripción más abajo), `-K` indica el valor de k-mero, `-p` indica el número de CPU, `-d` indica la frecuencia mínima de k-mero a considerar, `-o` indica el nombre de carpeta del proyecto.
- En la segunda línea donde se utiliza el comando contig se indica el nombre de la misma carpeta del proyecto `-g`.
- En la tercera línea donde se utiliza el comando map con `-s` se indica el fichero de configuración, con `-g` la carpeta del proyecto con `-K` el valor de k-mero, con –p el número de CPU.
- En la cuarta línea donde se utiliza el comando scaff, -g indica la carpeta del proyecto, `-F` permite rellenar los huecos en los *scaffolds* y `-p` el número de CPU.

También se puede ejecutar todas las etapas de forma seguida utilizando el comando `all`

```
SOAPdenovo-Trans-31kmer all -s config_file -K 29 -p 16 -d 1 -D 4 -F -o output
```

El fichero de configuración tiene el siguiente formato:

```
#longitude máxima de las lecturas
max_rd_len=150
[LIB]
#tamaño medio de inserto
avg_ins=180
#Si la secuencia se tiene que revertir, poner 0 para indicar que no
reverse_seq=0
#En que parte(s) utilizar la lectura (escoger 3 para utilizarla tanto en contigs como
en scaffolds)
asm_flags=3
#En qué orden las lecturas se utilizarán en el scaffolding (se considera en caso que
haya más de una librería de lecturas).
rank=1
# Número mínimo de pares de lecturas pareadas para realizar una conexión.
pair_num_cutoff=10
#Lecturas pareadas en formato fastq (utilizar q1 y q2)
q1=paired_reads1.fastq
q2=paired_reads2.fastq
#lecturas simples en formato fastq (utilizar q)
q=sequences_illum_454.fastq
```

Se pueden añadir más librerías escribiendo la etiqueta [LIB] seguida con los nombres de lecturas y sus características siendo el mismo modelo descrito arriba.

SOAPdenovo-trans fue utilizado para ensamblar los datos cortos de Illumina de *Tisochrysis lutea* y *Ruditapes decussatus*

III.3.2.7. CABOG

CABOG (Celera Assembler with Best Overlap Graph) [10, 54] es un ensamblador *de novo* de tipo OLC de secuencias largas (por ejemplo Sanger y 454) de ADN para genomas completos. CABOG ha contribuido de manera importante al avance de la genómica, incluyendo el primer ensamblaje completo un genoma de un organismo multicelular y el primer genoma diploide de un individuo humano.

El software de CABOG es un sistema modular compuesto de múltiples programas que interactúan a través de interfaces bien definidas de tal manera que se puede modificar el orden secuencias de los programas y sus parámetros para cubrir necesidades específicas.

La ejecución de CABOG se realiza con las dos líneas de comando siguientes con este mismo orden:

```
fastqToCA -insertsize 5000 1500 -libraryname my_library -technology '454' -type
'sanger' -innie -reads my_single_reads.fastq -mates my_paired_reads.fastq >
out_file.frg
runCA -d my_work_folder -p my_prefix -unitigger=utg out_file.frg
```

En la primera línea de comando `-insertsize` indica la información relativa a los tamaños de inserto a saber la media y la desviación típica (los dos valores separados por espacio), `-libraryname` indica el nombre de la librería de lecturas, `-technology` indica la tecnología de secuenciación de las lecturas, `-innie` indica que la orientación de las lecturas pareadas es "Forward-Reverse" (en caso que la orientación es "Reverse-Forward", se especifica `-outtie`), `-reads` indica el nombre del fichero .fastq de lecturas single, `-mates` indica el nombre del fichero de lecturas pareadas, `-out file.frg` indica el nombre de fichero de salida .frg. Las lecturas pareadas vienen intercaladas en el mismo fichero .fastq.

En la segunda línea de comando `-d` indica el nombre de la carpeta del trabajo, -p indica el prefijo de los ficheros generados, `-unitigger` indica el modulo utilizado para construir unitigs. CABOG ofrece tres módulos, de los cuales se utiliza solo uno a la vez. El modulo original (utg) es el más adecuado para datos de Sanger, el módulo "best overlap graph unitigger" (bog) es el más compatible con los datos de 454 y el módulo "bogart unitigger" (bogart) es el más compatible para los datos de Illumina. Al final de la línea de comando se añade el nombre del fichero .frg generado anteriormente.

CABOG fue utilizado para ensamblar las lecturas 454 de *Photobacterium damselae*.

III.3.2.8. Ray

Ray [68], es un ensamblador que utiliza el grafo de De Bruijn para construir su estructura de datos. La principal característica que podemos destacar de este ensamblador es que es un ensamblador que utiliza memoria distribuida para realizar sus operaciones. Los otros ensambladores disponibles, utilizan memoria compartida. Por lo tanto el ensamblador Ray utiliza MPI [117] como interfaz de paso de mensajes entre todos los nodos que estén funcionando de manera simultánea. Gracias a esta característica podemos aprovechar aún más el potencial del superordenador Picasso, y no tendremos que limitarnos a las 16 CPU que contienen los nodos de cálculo o las 64 CPU que contienen los nodos bigmem.

La ejecución del programa Ray se hace de la forma siguiente:

```
mpiexec -np 256 Ray -k 51 -p my_paired_reads1.fastq my_paired_reads2.fastq -s
my_single_reads.fastq -route-messages -connection-type debruijn -routing-graph-degree
4 -o illumina_01
```

donde –np indica el número de CPU utilizadas, –k indica el valor de k-mero, -p indica los nombres de los dos ficheros de lecturas pareadas separados por espacio, -s indica el nombre del fichero de lecturas simples.

El programa Ray tiene una amplia gama de configuraciones que se pueden consultar en `http://denovoassembler.sourceforge.net/manual.html`.

En caso que se trate de ensamblajes donde se utilizan grandes cantidades de lecturas, convendría activar los *routing messages*, añadiendo a la línea de comando lo siguiente:

```
-route-messages -connection-type debruijn -routing-graph-degree 4
```

El grado de los nodos de grafos utilizando la opción `-routing-graph-degree` según el número de núcleos que se van utilizar tal como viene explicado en la tabla III.1

Número de CPUs	Grado de nodos	Diámetro de nodos	Configuración
256	4	4 (4*4*4*4=256)	`-routing-graph-degree 4`
512	8	3 (8*8*8)	`-routing-graph-degree 8`
1024	4	5 (4*4*4*4*4)	`-routing-graph-degree 4`
1024	32	2 (32*32)	`-routing-graph-degree 32`
1024	2	10 (2^10)	`-routing-graph-degree 2`

Tabla III.1*: Configuración de los Grados de nodos De Bruijn en el programa Ray, según el número de CPUs utilizado*

Ray fue utilizado para ensamblar las secuencias de Illumina de *Solea seneganensis*

III.3.3. Para el tratamiento y mejora de los ensamblajes

III.3.3.1. GAM-NGS

En el ensamblaje del genoma de *Solea senelgalensis* (Apartado IV.3.2.2) era necesario reconciliar los contigs procedentes de las diferentes librerías de lecturas Illumina. Para ello se utilizó GAM-NGS [118] que es un programa que se utiliza para combinar dos o más ensamblajes genómicos con el fin de mejorar la contigüidad y la exactitud, y así obtener un ensamblaje único (de reconciliación) con mejores características que los originales. Las regiones que representan el mismo locus (llamados bloques) en los ensamblajes se identifican a través del mapeo de las lecturas y se guardan en un grafo ponderado. La fase de la combinación se lleva a cabo con la ayuda este grafo, lo que permite también la resolución de las regiones problemáticas.

Para reconciliar dos ensamblajes con GAM-NGS es necesario elegir uno de ellos como «maestro» (ensamblaje de base). El otro se definirá como «esclavo» y que se utilizará para complementar el primero.

Para reconciliar ensamblajes con GAM-NGS se requiere una etapa previa de mapeo de las lecturas sobre cada uno de ellos utilizando Bowtie2 [119], posteriormente se generará un fichero de alineamiento indexado en formato binario, BAM, utilizando samtools [120]. Estos dos últimos programas se describirán de forma más detallada en el apartado III.3.4. La etapa de mapeo de lecturas se realiza mediante las siguientes líneas de comando:

```
module load bowtie/2.2.4

bowtie2-build -f assembly_1.fasta my_indexes
```

```
bowtie2 my_indexes -q -1 paired_reads1.fastq -2 paired_reads2.fastq -U
single_reads.fastq --very-sensitive -p 8 -S assembly_1.sam

module load samtools

#convertir el fichero de format SAM a formato BAM
samtools view -bS assembly_1.sam > assembly_1.bam

#ordenar el fichero BAM
samtools sort assembly_1.bam sorted.assembly_1
#filtrar el fichero BAM (eliminar las lecturas no mapeadas)
samtools view -b -F 4 sorted.assembly_1.bam > filtered.sorted.assembly_1.bam

# crear un index al fichero BAM
samtools index filtered.sorted.assembly_1.bam
```

La ejecución del programa GAM-NGS se realiza en varias etapas:

```
PATH=`pwd -P`
THREADS_NUM=4
GAM_CREATE=gam-create
GAM_MERGE=gam-merge

# Etapa 1: Creacion de una carpeta donde guardar los ficheros de salida
mkdir -p ${PATH}/gam-ngs_merge

# Etapa 2: Preparacion de los ficheros de entrada

echo      -e      "${PATH}/Alignments/assembly_1/sorted.assembly_1.bam\n300      500"
>${PATH}/gam-ngs_merge/assembly_1.pe.list.txt
echo      -e      "${PATH}/Alignments/assembly_2/sorted.assembly_2.bam\n300      500”
>${PATH}/gam-ngs_merge/assembly_2.pe.list.txt


# Etapa 3: Construccion de los bloques

${GAM_CREATE} --master-bam ${PATH}/gam-ngs_merge/assembly_1.pe.list.txt --slave-bam
${PATH}/gam-ngs_merge/assembly_2.pe.list.txt --min-block-size 10 --output
${PATH}/gam-ngs_merge/out >${PATH}/gam-ngs_merge/gam-create.log.out 2>${PATH}/gam-
ngs_merge/gam-create.log.err

# Etapa 4: Reconciliacion

${GAM_MERGE} --blocks-file ${PATH}/gam-ngs_merge/out.blocks --master-bam ${PATH}/gam-
ngs_merge/assembly_1.pe.list.txt --master-fasta
${PATH}/Assembly/assembly_1/assembly_1.fasta --slave-bam
${PATH}/gam-ngs_merge/assembly_2.pe.list.txt --slave-fasta
${PATH}/Assembly/assembly_2/assembly_2.fasta --min-block-size 10 --output
${PATH}/gam-ngs_merge/out --threads ${THREADS_NUM}
>${PATH}/gam-ngs_merge/gam-merge.log.out 2>${PATH}/gam-ngs_merge/gam-merge.log.err
```

En la etapa 2 se genera en un archivo de configuración donde se imprimen los nombres de ficheros del mapeo con sus rutas así que el tamaño de inserto mínimo y máximo.

En la etapa 3, `--master-bam` indica el nombre del fichero BAM de mapeo de las lecturas sobre el ensamblaje considerado como maestro, `--slave-bam` indica el nombre del fichero BAM de mapeo de las lecturas sobre el ensamblaje considerado como esclavo.

En la etapa 4, `--master-fasta` indica el fichero de contigs o *scaffolds* del ensamblaje maestro, `--slave-fasta indica` indica el fichero de contigs o *scaffolds* del ensamblaje esclavo, `--min-block-size` indica el número de lecturas mínimo para considerar un bloque, `--output` indica el prefijo de los ficheros de salida, y con `--threads` se indica el número de cores a utilizar.

III.3.3.2. SOAPdenovo Scaffolder

En el ensamblaje del genoma de *Solea senegalensis* (Apartado IV.3.2.2) teníamos disponibles muchas librerías de lecturas que fueron aprovechadas para unificar los contigs. Uno de los programas que utilizamos para este propósito fue SOAPdenovo Scaffolder que realmente es un módulo del programa de ensamblaje genómico SOAPdenovo [116] que se utiliza para unir los contigs preensamblados con SOAPdenovo, sin embargo también puede usarse para unir contigs generados por otros ensambladores.

La ejecución del programa requiere una preparación previa de los contigs utilizando la herramienta finalFusion (`https://sourceforge.net/projects/soapdenovo2/files/Prepare/`), lo cual se hace con la siguiente línea de comando:

```
finalFusion -g Scaff -K 31 -c contigs.fasta -D
```

donde `-g` indica el prefijo de para la salida, `-K` indica el tamaño de k-mero, `-c` indica el fichero de contigs y con `-D` se activa el modo de preparación de los contigs. Se generaran varios ficheros que tienen el prefijo Scaff.

La unificación de los configs (*scaffolding*) se realiza con las siguientes líneas de comando:

```
# mapeo de las lecturas sobre los contigs
SOAPdenovo-63mer map -p 16 -s config -g Scaff 1>map.log 2>map.err

# unificacion de los contigs
SOAPdenovo-63mer scaff -p 16 -g Scaff -N 600000000 -F 1>scaff.log 2>scaff.err
```

donde `-p` indica el número de cores a utilizar, `-s` indica el fichero de configuración (véase el detalle de este fichero en el apartado III.3.2.6) `-g` indica el prefijo de los ficheros generados en la etapa de preparación, `-N` indica el tamaño esperado del genoma y con `-F` se activa el relleno de los huecos en los *scaffolds*

III.3.3.3. SSPACE

En los flujos de trabajo de los ensamblajes genómicos de *Photobacterium damselae* (Apartado IV.3.1.2) y *Solea senegalensis* (Apartado IV.3.2.2) era necesario unificar los *scaffolds* aprovechando de la información de las lecturas disponibles. Para ello utilizamos SSPACE [121] que es un programa independiente que sirve para el *scaffolding* de los contigs preensamblados mediante el uso de lecturas pareadas y tiene la ventaja de ser utilizado también para unir *scaffolds*. Este programa utiliza la información del mapeo de las lecturas pareadas sobre los contigs o *scaffolds* para evaluar el orden, la distancia entre ellos y la orientación por el fin de establecer conexiones entre los mimos.

La ejecución de SSPACE se realiza con una sola línea de comando de la siguiente forma:

```
SSPACE_Standard_v3.0.pl -l libraries.txt -s contigs.fasta -T 16
```

donde `-l` indica el nombre de un fichero de texto incluyendo la información sobre las lecturas pareadas que se utilizarán en la unificación de los contigs o *scaffolds*, `-s` indica el nombre del fichero de contigs o *scaffolds* de entrada, `-T` indica el número de CPU a utilizar.

El fichero libreries.txt tiene el siguiente formato:

```
lib1 bowtie lib1_paired_reads1.fastq lib1_paired_reads2.fastq 3000 0.25 FR
lib2 bowtie lib2_paired_reads1.fastq lib2_paired_reads2.fastq 8000 0.25 FR
```

donde la columna 1 corresponde el nombre de la librería, la columna 2 corresponde al programa utilizado para mapear las lecturas sobre los contigs o *scaffolds* en el cual se puede elegir Bowtie [119] o BWA [122] las columnas 3 y 4 corresponden a los nombres de ficheros de lecturas pareadas, las columnas 5 y 6 corresponden al tamaño de inserto de las lecturas pareadas y el error mínimo permitido en el tamaño de inserto respectivamente. Por ejemplo para un tamaño de inserto 3000 y un error de 0,25, la distancia entre las lecturas puede tener un error de 3000 * 0,25 = 750 en ambos sentidos. La última columna se refiere a la orientación de las lecturas pareadas (FR: directa-inversa, RF: inversa-directa, FF: directa-directa, RR: inversa-inversa)

III.3.3.4. SOAPdenovo GapCloser

SOAPdenovo GapCloser (`http://sourceforge.net/projects/soapdenovo2/files/GapCloser`) es un programa incluido en el paquete de SOAPdenovo que se utiliza para cerrar los huecos (indicado con N yuxtapuestas) que se crean durante la etapa de *scaffolding* con SOAPdenovo Scaffolder o con otro ensamblador. Está destinado a genomas grandes de plantas y animales aunque también funciona bien para genomas de bacterias y hongos.

La ejecución de este programa se realiza con la siguiente orden:

```
GapCloser -b config_file -a scaffolds.fasta -l 155 -t 64 -o
gap_closed_scaffolds.fasta
```

donde –b indica el nombre del fichero de configuración específico a SOAPdenovo (verse el detalle de este fichero apartado III.3.2.6), `-a` indica el nombre del fichero de *scaffolds* de entrada `-l` indica la longitud máxima de las lecturas (este valor puede tomar un valor máximo de 155), `-t` indica el número de cores a utilizar, y -o el nombre del fichero de salida.

SOAPdenovo GapCloser se utilizó en el flujo de trabajo de los ensamblajes genómicos de *Photobacterium damselae* y *Solea senegalensis*, para rellenar los huecos con nucleótidos para mejorar la calidad de los scaffolds.

III.3.4. Para analizar secuencias

III.3.4.1. SeqTrimNext

Antes de proceder al ensamblaje de las lecturas Roche/454 y Illumina tanto transcriptomicas como genómicas fue necesario preprocesarlas para descartar los fragmentos de baja calidad, contaminantes, vectores, adaptadores, y otros artefactos, para ello utilizamos SeqTrimNext. Se trata de una herramienta para el preprocesamiento de secuencias de nueva generación desarrollada en la Plataforma Andaluza de Bioinformática.

Se puede utilizar por línea de comandos, como servicio web basado en REST y como herramienta web. Para más información véase su portal `http://www.scbi.uma.es/seqtrimnext`.

A continuación se muestra un ejemplo de la ejecución de SeqTrimNext a través del sistema de colas de Picasso.

```
# se inicializa SeqTrimNext en el Cluster
. ~seqtrimnext/init_env
# se recogen las CPU asignadas.
srun hostname -s > workers
seqtrimnext -t my_template.txt -Q paired_seq1.fastq,paired_seq2.fastq -w workers
```

en la que `-t` indica que el fichero my_template.txt es una plantilla en la que se detallan los parámetros que se desean utilizar. En la versión web del programa se ofrecen plantillas con valores por omisión para varias situaciones (genómica, transcriptómica, amplicones, secuencias de plantas, etc.). El parámetro `-w` indica que el programa se ejecutará en los núcleos con las ID que se recogen en el fichero `workers`. Con `-Q` se indican los nombres de los ficheros con las secuencias (conviene consultar la ayuda del programa para ver más opciones y formatos de entrada).

III.3.4.2. Full-LengtherNext

Después de ensamblar los transcriptomas fue muy importante hacer una evaluación de su calidad. Para ello utilizamos Full-LengtherNext que es un programa originalmente diseñado para determinar si una EST se obtuvo a partir de un clon de ADNc que contiene todo el gen [123]. Solo estaba disponible vía web: `http://www.scbi.uma.es/cgi-bin/full-lengther/full-lengther_login.cgi`, donde tan solo requería un fichero con las secuencias (aceptaba varios formatos: fasta, phd, cromatogramas y ficheros comprimidos zip con cromatogramas), el valor de E mínimo para considerar como fiable un ortólogo, y el número máximo de aminoácidos que pueden faltar al principio del ortólogo. Más tarde, se desarrolló Full-LengtherNext, un programa nuevo adaptado a NGS y con más aplicaciones, como la de comprobar cuál es el mejor ensamblaje cuando las mismas secuencias se han ensamblado de distinta forma [87].

Full-LengtherNext también nos fue muy útil para seleccionar los programas y los kmeros utilizados para ensamblar las lecturas cortas según la calidad del transcriptoma generado.

III.3.4.3. AutoFact

Uno de los programas que utilizados para anotar los transcriptomas tratados en este trabajo fue AutoFact [83] que puede encontrarse en `http://www.bch.umontreal.ca/Software/AutoFACT.htm`. Una ventaja de este programa es que consulta numerosas bases de datos [UniRef90, UniRef100, NCBI's nr, COG, KEGG, Pfam, Smart, est_others, LSU (Large SubUnit ribosomal RNA), SSU (Small SubUnit ribosomal RNA)] para obtener la mejor descripción del producto del gen, clasificándolas de un modo estructurado, según el resultado encontrado, en:

Ribosomal RNA
Functionally annotated protein
Unassigned protein
[domain-name] containing protein
Unknown EST
Unclassified

En las ejecuciones realizadas en este trabajo se indicó en el fichero de configuración de AutoFact (AutoFACT.conf), que el orden de las bases de datos para obtener la información debía ser UniRef90, COG, kegg, Pfam, Smart, nr. es decir, las secuencias solo se anotan con información de EST, si previamente no se ha encontrado ninguna información fiable en el resto de bases de datos. Las bases de datos LSU y SSU no se utilizaron porque ya fueron incluidas en el preprocesamiento realizado con SeqTrimNext.

Para reducir el tiempo de cálculo de AutoFact se modificó la versión original del programa y se actualizaron las bases de datos (Guerrero et al., resultados no publicados).

Para ello se actualizó la versión de Blast que utilizaba el programa a la versión 2.2.20 de *Blastall*, que permite la paralelización del análisis. Además se modificó la línea de ejecución de Blast añadiendo un filtro de E = 10^{-3} y poniendo un número máximo de parecidos en 12. Anteriormente, AutoFact ejecutaba Blast sin limitación, lo que producía hasta 500 descripciones con sus 500 alineamientos por cada secuencia introducida, mientras que con la versión modificada se limitan únicamente a las 12 mejores, en el caso de que mejoren la probabilidad mínima indicada por el valor de E. Así se reduce el tiempo que requiere Blast para encontrar las secuencias y el tiempo necesario para que AutoFact analice los resultados de Blast, sin reducir la fiabilidad de éstos. Para ejecutar los trabajos de AutoFact se dividen las secuencias de entrada en paquetes de secuencias. Más pequeños son los paquetes, más núcleos son necesarios y más rápido es el trabajo. Los paquetes se lanzan por array job utilizando la opción ntasks para utilizar el máximo CUPs disponibles.

En el *array job*, después de añadir al fichero bash las líneas del código comentadas más arriba, se añaden estas líneas para lanzar los paquetes:

```
# inicializamos autofact
module load autofact

# ejecutamos el programa
AutoFACT.pl -f my_file_part${SLURM_ARRAYID}.fasta -a ../AutoFACT.conf -b -c 1
AutoFACT.pl -f my_file_part${SLURM_ARRAYID}.fasta -a ../AutoFACT.conf -p -c 1
```

En la salida de AutoFact, los ficheros con extensión .out contienen las anotaciones recogidas por cada base de datos y la seleccionada como la mejor por el algoritmo de AutoFact. Estos ficheros se utilizan posteriormente para importar las anotaciones de los unigenes a las bases de datos.

AutoFact fue uno de los programas utilizados para anotar los transcriptomas estudiados.

III.3.4.4. Basic Local Alignment Search Tool (BLAST)

El programa BLAST busca regiones similares entre secuencias. Compara secuencias de nucleótidos o proteínas con bases de datos de secuencias de nucleótidos o proteínas y devuelve las secuencias más parecidas en las bases de datos, ordenadas por el valor estadístico E, que proviene del valor de probabilidad p-value e indica el número de resultados esperados por azar con score tan alto como el obtenido. Las secuencias encontradas por Blast se consideran genes ortólogos a los que contienen nuestras secuencias de entrada. A lo largo de este trabajo se han utilizado muchas versiones de Blast [124], siendo ncbi-blast- 2.2.25+ [72] la última versión utilizada. Blast se puede descargar desde `ftp://ftp.ncbi.nlm.nih.gov/blast/ executables/blast+/LATEST/`.

Según el tipo de secuencias de entrada y la base de datos que haya que utilizar se ejecutará un Blast diferente. Durante la realización de este trabajo se utilizaron habitualmente:

BlastN: para comparar secuencias nucleotídicas con bases de datos de secuencias de nucleótidos. Este tipo de Blast fue muy utilizado en este trabajo, específicamente en el test de los ensambladores, en ICMapper y en la comparación de los transcriptomas de *Solea senegalensis* y *Solea solea*. También se utiliza en SoleaDB, IsochrysisDB, RuditapesDB.

BlastP: para comparar secuencias de proteínas con otras secuencias de proteínas. Como en este trabajo se trabaja principalmente con secuencias de nucleótidos de partida, este es el tipo de Blast fue menos utilizado. Pero fue útil para comparar ficheros de proteínas de distintas versiones/estados de organismos modelos.

BlastX: para comparar secuencias nucleotídicas con una base de datos de secuencias de proteínas. Este tipo de Blast fue utilizado para buscar ortólogos de los genes de las especies estudiadas en los organismos modelo. También esta utilizado en AutoFact, SoleaDB, IsochrysisDB, RuditapesDB.

TBlastN: para comparar secuencias de proteínas con una base de datos de secuencias de nucleótidos (esta base de datos se traduce a proteínas para llevar a cabo la comparación). Este tipo de Blast fue utilizado para buscar los porcentajes de proteínas de especies modelo entre los unigenes de *Tisochrysis lutea*. También es posible utilizarlo en la página web de SoleaDB, IsochrysisDB y RuditapesDB.

III.3.4.5. Sma3s

Para añadir más descripciones del producto del gen y más elementos de anotaciones en los transcriptomas estudiados utilizamos el programa Sma3s [84] que es un anotador de secuencias codificantes de proteínas escrito en Perl y usa módulos del proyecto BioPerl 1.6 (`http://www.bioperl.org`). Es accesible de forma gratuita en la web `http://www.bioinfocabd.upo.es/sma3s/`. Sma3s usa dos programas del paquete Blast: blastp para secuencias de aminoácidos y blastx para secuencias de nucleótidos usando un $E < 10^{-26}$, y blastclust para procedimientos de agrupamiento.

Para calcular los valores de probabilidad, Sma3s utiliza un módulo de enriquecimiento biológico basado en una distribución hypergeometrica (`http://www.cse.huji.ac.il/course/2006/bioskill/Ex2/HyGe.pm`).

Sma3s utiliza como su base de datos principal una base de datos UniProt en ficheros de formato texto plano .dat, correspondiente a la división taxonómica del organismo estudiado.

Sma3s se ejecuta de forma paralela en un *array-job*. Después de añadir al fichero bash las líneas del código comentadas más arriba, se añade la línea de comando de ejecución del Sma3s. Cada uno de los paquetes se lanza de la forma siguiente:

```
sma3s_v2.pl -a 123 -v 2 -d database_path/uniprot_vertebrates.dat -i
my_file_part${SLURM_ARRAYID}.fasta -b my_file_part${SLURM_ARRAYID}.blast -p F >
salida_${SLURM_ARRAYID}
```

III.3.4.6. RAST

Para anotar el genoma de la bacteria *Photobacterium damselae* (apartado IV.3.1.4) se utilizó RAST (Rapid Annotation using Subsystem Technology) [125] es un flujo de trabajo totalmente automatizado que está diseñado para anotar los genomas completos o borradores de genomas de bacterias y de arqueas, y que se ejecuta en el servidor web de RAST (`http://rast.nmpdr.org/`). El programa identifica los genes que codifican las proteínas, ARNr y ARNt, asigna funciones a los genes, predice los subsistemas representados en el genoma y utiliza esta información para reconstituir la red metabólica del organismo en cuestión, todo este conjunto de información se presenta en ficheros descargables. El genoma anotado puede ser consultado en un entorno que permite análisis comparativos con genomas anotados mantenidos en el entorno SEED (`http://www.theseed.org`).

III.3.4.7. Bowtie2

En muchas ocasiones fue necesario mapear las lecturas sobre unas secuencias de referencia que sea para saber la cobertura en lecturas sobre estas secuencias de referencia o para comprobar la fiabilidad de contigs o scaffolds, o también para la detección de los SNP. Para ello utilizamos Bowtie2 que es herramienta muy utilizada para el alineamiento de lecturas sobre secuencias largas de referencia, que pueden provenir del genoma o de los unigenes del transcriptoma de un organismo [119]. Se eligió este programa porque dispone de una buena documentación (`http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml`), tiene una gran cantidad de parámetros ajustables para su ejecución, es capaz de trabajar en paralelo, sale bien parado cuando se compara con otros mapeadores [126] y sobre todo porque es muy rápido. A continuación se describe brevemente cómo se ha utilizado Bowtie2 en este trabajo, teniendo en cuenta que se ejecuta en dos pasos. En un primer paso, a partir de las secuencias de referencia guardadas en el fichero multifasta `my_reference.fasta`, crea los índices de la referencia en el fichero `my_indexes` del siguiente modo:

```
bowtie2-build -f my_reference.fasta my_indexes
```

En la siguiente etapa se emplea el fichero `my_indexes` para alinear las lecturas contenidas en el fichero my_reads_1.fastq a la referencia:

```
bowtie2 my_indexes -q -1 my_paired_reads1.fastq -2 my_paired_reads2.fastq -U
my_single_reads_1.fastq -p 8 -S my_align.sam
```

donde `-q` indica que el fichero de lecturas está en formato FASTQ [127], `-U` indica que lo que viene a continuación es el nombre del fichero o ficheros (separados por comas) con las lecturas, `-p` indica el número de CPU a utilizar, y `-S` el nombre del fichero de salida con los alineamientos en formato SAM [120].

III.3.4.8. Samtools

Para parsear y manipular alineamientos en el formato SAM y BAM se utilizó Samtools [120] que es un paquete de software y librerías disponible en `http://samtools.sourceforge.net/`

Una de las tareas en las cuales se utilizó el samtools fue convertir ficheros de alineamiento SAM en su forma binarioa BAM, un fichero con un tamaño mucho más pequeño que el original en formato SAM, con lo cual, para ahorrar espacio conviene convertir los ficheros de formato BAM antes de guardarlos. Esta conversión se hace con la herramienta `view` y se ejecuta de la forma siguiente:

```
samtools view -bS my_alignment.sam > my_alignment.bam
```

Samtools se utilizó también para ordenar los alineamientos basándose sobre las secuencias de referencia. Este paso es necesario para poder visualizarlos con el programa Tablet por ejemplo. La operación de ordenamiento se hace de la forma siguiente:

```
samtools sort mi_alignment.bam sorted.mi_alignment
```

Otras funcionalidades del samtools utilizada en este trabajo, es el descubrimiento de los SNP, específicamente cuando los datos son secuencias de lecturas cortas (Illumina) u una combinación de secuencias de Illumina y 454/Roche.

Para descubrir los SNP en un organismo hay que tener las lecturas ya ensambladas con uno de los programas de ensamblaje de tipo De Bruijn. Después se utiliza el programa Bowtie2 para crear un fichero de alineamiento .SAM de las lecturas sobre los contigs generados previamente en el ensamblaje. Desde entonces se utilizan varias herramientas del samtools sucesivamente para tratar el fichero de alineamiento. Aquí presentamos las diferentes etapas del tratamiento del fichero .SAM hasta la generación el fichero .VCF (Variant call format) [128]:

(i) Convertir el fichero .SAM a .BAM de la misma forma que se comentó anteriormente.

```
samtools view -bS mi_alignment.sam > mi_alignment.bam
```

(ii) Ordenar los alineamientos por referencia de la misma forma que se comentó anteriormente.

```
samtools sort mi_alignment.bam sorted.mi_alignment
```

(ii) Filtrar los alineamientos (eliminar las lecturas que no mapearon):

```
samtools view -b -F 4 sorted.my_alignment.bam > filtered.sorted.my_alignment.bam
```

(iv) Generar un fichero binario .BCF

```
samtools mpileup -C50 -d5000 -I -uf my_unigenes.fasta
filtered.sorted.my_alignment.bam | bcftools view -bvcg - > my_alignment.bcf
```

(v) Generar el fichero final .VCF

```
bcftools view my_alignment.bcf | vcfutils.pl varFilter -D5000 > my_alignment.vcf
```

Las herramientas bcftools y vcfutils.pl están incorporadas en el paquete de samtools.

III.3.4.9. CD-HIT

En los flujos de trabajo de los ensamblajes de transcriptomas era necesario agrupar los contigs procedentes de los ensamblajes con múltiples kmeros de lecturas cortas a fin de eliminar las redundancias y disminuir el número de secuencias de entrada en la etapa de reconciliación. Lo cual fue posible gracias a CD-HIT que es un programa para el agrupamiento de secuencias, tanto de nucleótidos como de proteinas [129], disponible en `http://cd-hit.org`. CD-HIT agrupa las secuencias según unos criterios indicados y devuelve una secuencia consenso por cada conjunto de secuencias que se parecen entre sí. Ejemplo de ejecución para agrupar secuencias de nucleótidos:

```
cd-hit-est -i input.fasta -o output.fasta -c 0.9 -T 4 -M 6000
```

donde `-i` y `-o` indican los ficheros de entrada y salida respectivamente, `-c` el porcentaje mínimo de similitud entre las secuencias que se agrupan, `-T` indica el número de CPU utilizadas, y `-M` la memoria RAM utilizada.

Para más información acerca de cómo ejecutar el programa y de sus parámetros es recomendable consultar su manual (`http://www.bioinformatics.org/cd-hit/cd-hit-user-guide.pdf`).

III.3.4.10. MREPS

La detección de los marcadores SSR en los transcriptomas se realizó con MREPS que es un programa que detecta las repeticiones de secuencias simples a partir de un fichero en formato fasta [130]. Este programa tiene la ventaja de ser simple y muy rápido. Disponible en: `http://bioinfo.lifl.fr/mreps/`, la versión que se ha utilizado es la 2.5. Se ejecuta con la orden

```
mreps -minsize 12 -minperiod 2 -exp 3.0 -fasta my_file.fasta
```

donde `-minsize` indica la longitud mínima de la repetición encontrada, `-minperiod` el número mínimo de nucleótidos que se repiten (2:GAGA 3:CATCAT 4:GATAGATA), y `-exp` el número mínimo de veces que se repite (2:TATA 3:TATATA 4:TATATATA).

III.3.5.11. Gigabayes

Para detectar las variaciones genéticas SNP en los ensamblajes de las lecturas Roche/454 se utilizó Gigabayes [131] que es un programa optimizado para el análisis de millones de lectura NGS alineados a una secuencia consenso que puede ser un transcrito o un ADN genómico.

El descubrimiento de los SNP con Gigabayes se hace a través del análisis de un fichero .ACE que es un fichero de alineamiento de las lecturas sobre la secuencia del consenso, el cual lo generan la mayoría de los ensambladores OLC.

El análisis de Gigabayes se hace en dos etapas. En la primera etapa se crea el fichero binario a partir de los datos de alineamiento de las lecturas y en la segunda se analiza la información contenida en este fichero binario.

```
gigaBuild --fd my_file.fasta --fq my_file.qual --ace ./my_file.ace --gig
my_file.ace.gig

gigaBayes --gig my_file.ace.gig --gff newfile.ace.gff --anchor --ploidy diploid --
indel --debug --CRL 4 --PSL 0.9 --log my_file.ace.log
```

- En la primera línea de comando `--fd` indica el fichero de lecturas utilizado en el ensamblaje, `--fq` indica el fichero de calidades, `--ace` indica el fichero de alineamiento y `--gig` indica fichero binario de salida.

- En la segunda línea de comando `--gig` indica fichero binario generado anteriormente, `--gff` indica un fichero de tipo .gff que se general durante el análisis, `--anchor` indica la utilización de secuencia de anclaje para buscar el polimorfismo. `--ploidy` se refiere a la ploidía del organismo (“haploid” o ”diploid”), la opción `--debug` permite que se muestren mensajes de depuración del programa. `--CRL` indica la cobertura mínima en una posición para ser considerada. `--PSL` indica el valor mínimo de probabilidad de un SNP para que se muestre en el informe, `--log` indica el fichero de salida donde se imprime toda la información relativa a los SNPs encontrados. Para incluir en el análisis la búsqueda de los Indels se añade la opción `--indel`.

Gigabayes se utilizó para descubrir los SNP en los proyectos de ensamblaje de *Solea senegalensis* donde se utilizaron únicamente las lecturas de 454/Roche.

III.3.4.12. Tablet

Se trata de una herramienta gráfica interactiva para la visualización de ensamblajes [132]. Admite multitud de formatos diferentes: ACE, AFG, MAQ, SOAP2, SAM, BAM, FASTA, FASTQ y GFF3.

Puede descargarse en http://bioinf.scri.ac.uk/tablet/ y es muy sencillo de utilizar, tan solo hay que abrir el programa y elegir el fichero de ensamblaje que se quiere visualizar.

Tablet se utilizó para la visualización de los ensamblajes de lecturas Roche/454 y para observar las corrección aportadas por el programa CoMiner (Apartado IV.1.3)

III.3.4.13. GEvo

Para poder comprobar la colinealidad de los genomas de *Solea senegalensis* y *Cynoglossus semilaevis* (Apartado V.3.2.3.2), usamos GEvo (`https://genomevolution.org/CoGe/GEvo.pl`) [133] que es una herramienta web diseñada para comparar regiones genómicas entre múltiples organismos utilizando diferentes algoritmos de comparación, lo cual permite de identificar patrones en la evolución de los genomas.

III.3.4.14. NUCMER

En el ensamblaje del genoma de *Solea senegalensis* (*Apartado* IV.3.2.2), era necesario alinear los contigs por el fin de combinar aquellos que solapan entre ellos. Para ello se utilizó NUCMER que es un programa del paquete MUMmer [134] muy utilizado el alineamiento global de genomas enteros que sean completos o en estado de borrador y que tiene la ventaja de ser muy rápido en sus tareas de alineamiento. Se ejecuta con los dos siguientes ordenes sucesivos:

```
nucmer -p nucmer reference_genome.fasta query_genome.fasta

show-coords -T nucmer.delta > nucmer.coords
```

En la primera línea, reference_genome.fasta indica el nombre del fichero fasta que incluye al genoma escogido como referencia, query_genome.fasta indica el nombre del fichero fasta que incluye al genoma en el cual se buscarán alineamientos con el primer genoma. Con esta línea se genera el fichero nucmer.delta

III.3.5. Scripts escritos

En este apartado se describen los scripts que fueron escritos a lo largo de este trabajo para tratar y procesar los datos genómicos de las especies estudiadas.

III.3.5.1. Para generar lecturas artificiales

Este script escrito en perl (apéndice A), se utilizará en el apartado IV.1.2 para generar lecturas artificiales por el fin de testear los programas de ensamblaje. Se ejecuta con la orden:

```
generate_artif_rd.pl sequences.fasta 300 500 30
```

donde el primer parámetro representa el nombre del fichero de secuencias de donde se extraerán las lecturas artificiales, el segundo parámetro representa la longitud mínima de las lecturas, el tercer parámetro la longitud máxima y el cuarto parámetro la cobertura de las lecturas.

Este script se utilizó para crear secuencias artificiales para utilizarlas en la verificación de los programas de ensamblaje de lecturas largas.

III.3.5.2. Para combinar dos ficheros ACE

Es un script en Ruby que combina dos ficheros ACE de MIRA y CAP3 (ver apartado IV.1.2.3 más adelante). Utiliza la gema scbi_ace (apartado III.3.1.2) y se ejecuta de la siguiente manera:

```
combine_ace_files.rb assembly_cap3.ace assembly_mira.ace mira_reads_index
```

donde el primer parámetro representa el nombre del fichero ACE de CAP3 el segundo parámetro representa el nombre del fichero ACE de MIRA, y el tercer parámetro el índice de los identificadores de secuencias de MIRA

Este script se utilizó para combinar los ficheros ACE de MIRA y CAP3 en los ensamblajes de algunas versiones del transcriptoma Solea senegalensis donde solo se utilizarón lecturas 454/Roche.

III.3.5.3. Para calcular el tamaño de inserto real en lecturas pareadas.

Este script escrito en Ruby (apéndice B) utiliza el fichero SAM de mapeo de una librería de lecturas sobre una referencia (que puede ser un ensamblaje preliminar de estas lecturas formado de contigs) para calcular el tamaño de inserto real de esta librería. El script se utiliza de la siguiente forma:

```
insert_size_from_sam.rb mapping_file.sam
```

en el que `mapping_file.sam` representa el nombre del fichero de mapeo previamente mencionado.

Este script fue utilizado para calcular el tamaño de inserto real en las lecturas pareadas (Roche/454 o Illumina) utilizadas en este trabajo.

III.3.5.4. Para eliminar ensamblajes erróneos

Este script escrito en Ruby recorre el resultado de alineamiento Blast entre los transcritos procedentes de lecturas cortas con los procedentes de las lecturas largas (considerados como más fiables) y entre los transcritos procedentes de lecturas cortas detecta aquellos que tienen una estructura sentido/antisentido para poder eliminarlos. Se ejecuta de la siguiente forma:

```
misassembled_trascripts.rb blast_alignment 95 80
```

donde el primer parámetro indica el fichero Blast de alineamiento, el segundo indica el porcentaje de identidad mínimo para considerar un alineamiento, y el tercero la longitud de alineamiento mínimo.

Este script se utilizó en los flujos de trabajo de los ensamblajes de transcriptomica para eliminar los ensamblajes erróneos en los contigs procedentes de los ensamblajes illumina.

III.3.5.5. Para seleccionar los transcritos que representan las sondas de microarray.

Es un script en Ruby que utiliza el resultado de Full-LengtherNext y el de dos predictores de secuencias codificantes: testcode [135] y ORF predictor [136] (`http://proteomics.ysu.edu/tools/OrfPredictor.html`), para seleccionar automáticamente los transcritos que representarán las sondas para un experimento de microarrays. La ejecución se realiza de la siguiente forma:

```
fln2microarray.rb fln_annotation_file test_code_file orf_predictor_file
transcripts.fasta 44000
```

donde `fln_annotation_file` indica el nombre del fichero de anotación de Full-LengtherNext, `test_code_file` indica el nombre del fichero la salida de testcode, `orf_predictor_file` indica el nombre del fichero de salida del programa ORF predictor, transcripts.fasta indica el nombre del fichero fasta de transcritos y con el ultimo parámetro se indica el número de transcritos requeridos para el microarray.

Este script se utilizó para seleccionar los transcritos de *Solea senegalensis* que se enviaron para realizar un experimento de microarrays

III.3.5.6. Para calcular las estadísticas sobre los marcadores SSR

Es un script escrito en Ruby (apéndice C) que se utiliza para recorrer el fichero de salida de MREPS (apartado III.3.4.10) que sirve para de detección de marcadores SSR en un transcriptoma, y desde ello calcula estadísticas sobre estos marcadores tanto los que están dentro del ORF como aquellos que están en la UTR.

La ejecución de este script se realiza de la siguiente forma:

```
ssrs_stats.rb mreps_file transcripts_ids fln_pt_seqs
```

donde `mreps_file` indica el nombre del fichero de salida de MREPS, `transcripts_ids` indica el nombre de un fichero que incluye a los identificadores de transcritos (un id por línea) de los cuales se obtendrán las estadísticas de los SSR incluidos en los mismos. `fln_pt_seqs` indica el nombre del fichero de anotación de Full-LengtherNext

Este script se utilizó para extraer las estadísticas sobre los marcadores SSR detectados en los transcriptomas estudiados.

III.3.5.7. Para solapar los contigs genómicos

En los ensamblajes genómicos, los contigs pueden compartir zonas comunes en sus bordes. Éstas se pueden detectar con un programa de alineamiento global como NUCMER (apartado III.3.4.14).

Con este script en Ruby se utiliza el resultado de NUCMER para solapar los contigs según unos criterios bien definidos y utilizando la información de sintenia con ICMapper (ver apartado III.3.5.9 a continuación) de dos especies cercanas diferentes. Globalmente, hay dos criterios para solapar los contigs: sea que tengan unos alineamientos muy estrictos (longitud y porcentaje de identidad del alineamiento muy altos), o que estos parámetros sean mínimos y que los dos contigs se encuentran contiguos sobre la base de la sintenia con especies cercanas.

Después de obtener el fichero de coordenadas del alineamiento del genoma consigo mismo utilizando NUCMER, la ejecución del script se realiza de la siguiente forma:

```
contigs_ovlp.rb -n nucmer.coords -l 100 -i 80 -fr species_1.ordering -sr
species_2.ordering -c contigs.fasta
```

donde –n indica el nombre del fichero de coordenadas que representan el alineamiento de NUCMER. –l indica la longitud mínima inicial de alineamiento (filtro preliminar), -i indica el porcentaje de identidad inicial mínimo (filtro preliminar), -fr y –sr indican la información de ordenamiento sobre la base de la sintenia con dos especies cercanas. Con –c se indica el nombre del fichero de contigs en los cuales se realizaran los solapamientos.

Este script fue utilizado para combinar los contigs solapantes del genoma de *Solea senegalensis.*

III.3.5.8. Para la corrección automática de los ensamblajes (Cominer)

Este programa esta descrito de forma detallada más adelante en el apartado IV.1.3. Su ejecución se realiza de esta forma:

```
cominer.rb -a assembly.ace -t 40 -e 2 -l 40 -p
```

donde `-a` indica el fichero ACE a analizar `-t` indica la longitud máxima a recortar en porcentaje de la lectura, `-e` indica el porcentaje mínimo de los errores en las lecturas para esta sea considerada. `-l` indica la longitud mínima de lecturas a considerar en el ensamblaje nuevo, `-p` es para generar un plot sobre la zonas de distribución de los errores.

III.3.5.9. Para ordenamiento de los contigs o scaffolds y acabado de los genomas (ICMapper)

Este programa escrito en Perl ordena los contigs o scaffolds de un borrador de genoma basándose en una referencia; esta descrito de forma detallada más adelante en el apartado IV.1.4; ICMapper en se ejecuta en dos etapas:

Alineamiento Blast entre los contigs o scaffolds y un genoma de referencia acabado:

```
blastn -task 'blastn' -db reference.fasta -query scaffolds.fasta -dust
no -culling_limit 1 -outfmt '6 qseqid sacc pident length qstart qend sstart send qlen
slen sstrand' > result.blast
```

Ordenamiento de los contigs o scaffolds:

```
icmapper.pl -f result.blast -m 100 -d 7000 -g 10000
```

donde `-f` indica el nombre del fichero de salida del alineamiento Blast entre el borrador de genoma y la referencia, `-m` indica el alineamiento mínimo a considerar, `-d` y `-g` respectivamente indican la separación máxima de las diagonales donde se ubican los HSP y la longitud máxima de los huecos entre los HSP (debajo de estos dos parámetros, los HSP se pueden considerar dentro del mismo cluster).

Este programa se utilizó en el ensamblaje del genoma de *Solea senegalensis* donde nos fue útil para confirmar la contigüidad de los contigs solapantes y los scaffolds unificados tomando como referencia el genoma de *Cynoglossus semilaevis*. También se utilizó para construir los *super-scaffolds* de *Solea senegalensis* sobre la base de los cromosomas de *Cynoglossus semilaevis*

III.4. Datos biológicos

III.4.1. Secuencias transcriptómicas

III.4.1.1. Para el transcriptoma de *Solea senegalensis*

Los datos utilizados consistieron en lecturas largas de 454/Roche y lecturas cortas de Illumina.

Lecturas 454/Roche: 14 librerías simples, entre ellas 12 corresponden a 3 zonas del organismo que son testículo/ovario, hipófisis y el hipotálamo, una al sistema inmunitario y una al sistema de osmoregulación. El número total de lecturas fue 5 663 225 con una longitud media de 757 nt

Lecturas Illumina: Esta lecturas provienen de experimentos de RNAseq que incluyen a un experimento temporal donde se obtuvieron muestras en los diferentes estadios de desarrollo de *S. senegalensis* (huevos, embriones, larvas, y estadio metamórfico) y otros experimentos de incubación en diferentes sustancias: AR (ácido retinoico), ATRA (all-trans AR), DEAB (4-diethyl amino-benzaldehyde), DMSO (dimethyl sulfoxide) y TTNPB (agonista de rars) . Se seleccionó una muestra de cada condición experimental para formar un total de 37 muestras que corresponden a lecturas cortas pareadas de Illumina de longitud inicial 2 x 76 nt.

III.4.1.2. Para el transcriptoma de *Solea solea*

Para este experimento las lecturas utilizadas son exclusivamente de Illumina obtenidas de un experimento de RNAseq donde se usaron tratamientos análogos a los del experimento de RNAseq de *S. senegalensis* (apartado anterior) y consisten en 43 librerías de lecturas tipo pareadas de longitud inicial 2 x 100 nt.

III.4.1.3. Para el transcriptoma de *Tisochrysis lutea*

Se utilizaron tanto datos de 454/Roche como de Illumina, y las características de las lecturas por cada librería se muestran en tabla III.2.

Tabla III.2: *Características de las lecturas brutas utilizadas en el ensamblaje del transcriptoma de Tisochrysis lutea*

Tecnología	Índice de la librería	Nº de librerías	Normalización	Tipo	Tamaño de inserto (nt)	Longitud de lecturas (nt)	Nº de lecturas
Illumina	1	3	Si	PE	150	2 x 100	925 682 496
	2	7	No	PE	150	2 x 76	612 903 376
454	1	2	Si	SE	-	547	1 151 067
	2	1	No	SE	-		505 781

III.4.1.4. Para el transcriptoma de *Ruditapes decussatus*

Se utilizó una sola librería de 127 327 692 lecturas Illumina pareadas de tipo PE y de longitud inicial 2 x 75 nt.

III.4.2. Secuencias genómicas

III.4.2.1. Secuencias genómicas de *Photobacterium damselae* subsp. *piscicida*

Se utilizaron exclusivamente secuencias de 454/Roche (tabla III.3)

Tabla III.3: *Características de las lecturas brutas utilizadas en el ensamblaje de Photobacterium damselae subsp. piscicida*

Cepa	Tecnología	Índice de las librerías	Nº de librerías	Tipo	Tamaño de inserto	Longitud de lecturas (nt)	Nº de lecturas
L091106-03H	454	1	1	MP	8 kb	509	148 622
		2	1	SE	-	1 195	297 269
DI21	454	1	2	MP	8 kb	445	433 717
		2	1	SE	-	550	187 433

III.4.2.2. Secuencias genómicas de *Solea senegalensis*

En este ensamblaje se utilizaron tanto lecturas de 454/Roche como de Illumina, las características de las diferentes librerías utilizadas se ilustran en la tabla III.4

Tabla III.4: Características de las lecturas brutas utilizadas en el ensamblaje del genoma de *Solea senegalensis*

Tecnología	Índice de las librerías	Nº de librerías	Tipo	Sexo	Tamaño de inserto	Longitud de lecturas (nt)	Nº de lecturas
Illumina	1	27	PE	Hembras	350	2 x 100 2 x 152	869 031 542
	2	6	PE	Hembras	300	2 x 100	1 005 527 592
	3	12	MP	Hembras	3 kb	2 x 100	1 109 021 114
454	1	4	MP	Hembras	3 kb	551	2 103 984
	2	9	MP	Hembras	8 kb	546	5 236 315
	3	1	MP	Hembras	20 kb	579	1 018 187

Parte IV
Resultados y discusión

IV. Resultados y discusión

IV.1. Algoritmos preparatorios.

En este trabajo, nuestro objetivo general es de secuenciar y ensamblar tanto secuencias genómicas como transcriptómicas de lenguado y especies afines. Sin embargo, la obtención de los resultados optimos dependía de la elección de las herramientas bioinformáticas más adecuadas para realizar tareas definidas del analisis bioinformático. Para obtener los resultados optimos, convenía tener una visión general de los algoritmos más ilustrativos para cada uno de los casos y así poder escoger cuál da los mejores resultados con secuencias conocidas antes de pasar a las secuencias desconocidas. Puesto que estos ensamblajes se harían con secuencias (principalmente 454) recién obtenidas por otros laboratorios de investigación con los que colaboramos, también teníamos que tener a punto el programa de preprocesamiento de las secuencias brutas.

IV.1.1. Definición de nuevas especies contaminantes

En cualquier laboratorio se convive con una gran cantidad de microorganismos que pueden acabar contaminando una muestra. Por ejemplo, *Delftia* es un contaminante frecuente en los reactivos [137]. También se consideran contaminantes los restos de tejidos humanos procedentes de los investigadores, así como las secuencias de ADN de los microorganismos utilizados con frecuencia en el laboratorio, como *E.coli* [42] y *Agrobacterium tumefaciens* [43]. Los hongos del polvo domestico también pueden contaminar muestras, siendo los más frecuentes *Penicillum* sp., *Aspergillus* sp. y las levaduras [138]. Otros tipos de contaminantes son los orgánulos del organismo de estudio (cloroplastos y mitocondrias) y las poblaciones microbianas que crecen sobre dicho organismo [31]

En nuestro laboratorio se ha desarrollado el programa SeqTrimNext (`http://www.scbi.uma.es/seqtrimnext/`; [46]) para el preprocesamiento de las lecturas brutas obtenidas por ultrasecuenciación. Está dotado de una base de datos de genomas de organismos contaminantes en potencia, como bacterias, hongos, además de orgánulos y ARNr.

En este trabajo vamos a trabajar con lenguado como especie principal. Las características de su crianza y mantenimiento hacen que, además de los contaminantes convencionales, hay que añadir nuevas especies, sobre todo las utilizadas en su régimen alimentario. Así, incluimos también entre los contaminantes las secuencias publicadas tanto genómicas (cromosomas nucleares y mitocondriales) como transcriptómicas (transcritos y EST) obtenidos del NCBI de las siguientes especies:

Artemia salina
Artemia parthenogenetica
Artemia francescana
Brachionus plicatilis
Isochrysis galbana

IV.1.2. Selección de las condiciones de ensamblaje *de novo*

Los programas de ensamblaje se enfrentan al problema de las secuencias repetidas, que provocan ensamblajes erróneos o interrumpidos (fragmentados). Las diferentes regiones genómicas pueden compartir repeticiones perfectas e indistinguibles. El problema es especialmente irresoluble cuando la repetición es más larga que la longitud de las lecturas. Puesto que del lenguado y las especies afines no había información al comienzo de esta tesis, todo el desarrollo tenía que hacerse desde cero.

Lo primero que hicimos fue comparar el rendimiento de cinco programas de ensamblaje (tabla IV.1), dos de ellos utilizan variantes de la estrategia de extensión por solapamiento (*Overlap-Layout-Consensus*, OLC), y los demás utilizan el algoritmo de grafos de De Bruijn que se recorren con caminos eulerianos. Estos programas tienen parámetros que permiten configurarlos tanto para genómica como para transcriptómica, excepto el programa CABOG, que es exclusivo para ensamblajes genómicos.

Tabla IV.1*: Programas comparados en el estudio*

Programa	Tipo	Referencia	URL
CAP3	OLC	[105]	http://seq.cs.iastate.edu/
MIRA	OLC	[139]	http://sourceforge.net/projects/mira-assembler
CABOG	OLC	[54]	https://wgs-assembler.sourceforge.net
VELVET	De Bruijn	[11]	https://www.ebi.ac.uk/~zerbino/velvet/
EULER	De Bruijn	[65]	-

IV.1.2.1. Ensamblaje de transcriptomas

En primer lugar evaluamos la capacidad de los programas CAP3, MIRA, VELVET y EULER para ensamblar transcriptomas a partir de lecturas artificiales extraídas a partir de transcriptomas conocidos y después se compararon las características de los contigs generados por cada uno de ellos. En la figura IV.1 aparecen las etapas clave del método utilizado.

Se descargaron cuatro transcriptomas de diferentes especies que fueron 2 vegetales (Arabidopsis thaliana y Zea mays, descargados desde NCBI) y 2 peces (*Oreochromis niloticus* y *Danio rerio,* descargados desde Ensembl). De cada uno se extrajo una colección de transcritos que contiene un número más reducido de secuencias a fin de disminuir el requerimiento de algunos programas en tiempo y memoria, sobre todo en coberturas altas en lecturas (tabla IV.2).

Para la creación de las lecturas artificiales diseñamos un script Perl que, a partir de una colección de transcritos, generaba lecturas de longitud aleatoria dentro de un intervalo

definido y con una cubertura definida. En este experimento se generaron lecturas artificiales de 300 a 500 nt para simular las lecturas producidas por la tecnología 454/FLX Titanium. También se generó un fichero de calidad ya los programas MIRA y VELVET lo requieren en sus ensamblajes. Se escogieron valores fijos de 60 para descartar el efecto de la variación de calidad de los nucleótidos en estos ensamblajes ya que en el ensamblaje con EULER no se utilizan los valores de calidad.

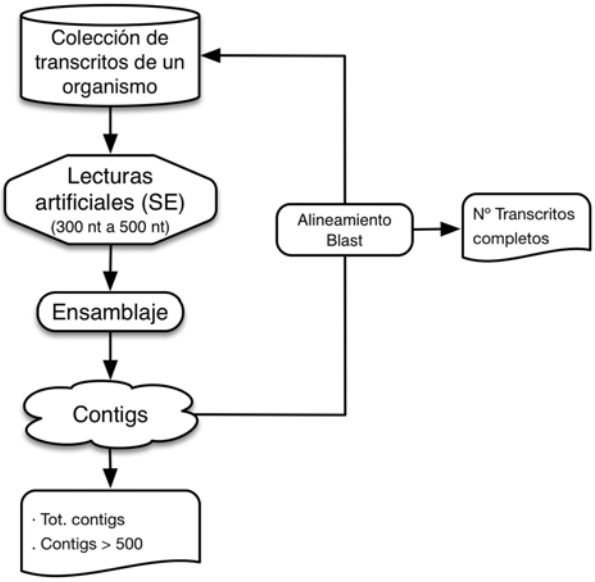

***Figura IV.1**: Esquema del protocolo utilizado en el testeo de un programa de ensamblaje de lecturas transcriptómicas*

***Tabla IV.2**: Material utilizado en la comparación de los programas de ensamblaje*

Organismo	*Nº de transcritos utilizados*	*Número de lecturas artificiales (30x)*
Arabidopsis thaliana	*5 215*	*690 065*
Zea mays	*3 382*	*366 086*
Oreochromis niloticus	*3 399*	*650 840*
Danio rerio	*5 000*	*733 238*

A partir de los contigs (transcritos) generados por los programas, se recopilaron unos datos básicos que son el número total de contigs y el número de contigs superiores a 500 nt. Para valorar la calidad de los contigs, se calculó el porcentaje de la secuencia original que había logrado reconstruir el contig en función de la longitud de alineamiento con Blast entre el contig y el transcrito original correspondiente (figura IV.1). En la tabla IV.3 se resumen los resultados obtenidos por cada especie y por cada programa de ensamblaje, indicando el número acumulado de contigs que recuperaban >80% de la secuencia original, >85%, >90%, >95% y el 100% de las mismas (columnas "Completitud de los transcritos" en la tabla IV.3).

Se presentan los datos de la cobertura 30X, que era la más baja en la que se estabilizaban los resultados, esto es, con coberturas de 60X no se observaban cambios significativos. Tal como se observa, todos los programas generaron un número de contigs superior al número de transcritos originales, lo que muestra un cierto nivel de fragmentación y redundancia. Esta abundancia de contigs fue más pronunciada en el caso VELVET y EULER específicamente en su ensamblaje de las dos especies de peces (tabla IV.3, C y D), aunque en estos casos el número de contigs superiores a 500 fue mucho más reducido que el número total, lo que indica una presencia de muchos fragmentos pequeños o muy pequeños seguramente debido al tipo de algoritmo. En caso de CAP3 y MIRA el número de contigs se acercaba más al número de transcritos originales y la mayoría de los contigs fueron largos. Por tanto, parece que los algoritmos OLC reconstruyen mejor el número de transcritos que los De Bruijn.

En lo que se refiere a la completitud de los transcritos, VELVET fue el programa que generó más contigs parecidos a los originales (>80%), posiblemente porque muchos de ellos no llegaron a generar el transcrito completo (100%). No obstante, MIRA y EULER se distinguieron de los demás por generar un alto número de contigs completos a 100%. Estos dos programas, el primero de tipo OLC y el segundo de grafos de De Bruijn, parecen ser los mejores candidatos para una estrategia combinatoria de ensamblaje de transcriptomas.

***Tabla IV.3**: Características de los transcritos producidos por cada programa de ensamblaje de novo de transcriptomas*

A. Arabidopsis thaliana

Nº de transcritos originales: 5 215

Nº de lecturas artificiales (30x): 690 065

	Nº transcritos		Completitud de los transcritos				
Programa	Totales	> 500n	>80%	>85%	>90%	>95%	100%
CAP3	5 618	5 506	2 287	2 198	2 084	1 900	1 536
MIRA	5 541	5 181	4 987	4 910	4 747	4 429	3 430
VELVET	5 399	5 158	5 097	5 087	5 032	4 028	0
EULER	6 618	5 119	4 883	4 856	4 834	4 778	4 502

B. Zea mays

Nº de transcritos originales: 3 382

Nº de lecturas artificiales (30x): 366 086

	Nº transcritos		Completitud de los transcritos				
Programa	Totales	> 500n	>80%	>85%	>90%	>95%	100%
CAP3	4 501	4 204	2 306	2 227	2 139	2 002	1 675
MIRA	3 356	3 243	3 099	3 081	3 027	2 874	1 427
VELVET	3 582	3 248	3 156	3 138	3 079	2 346	4
EULER	4 400	3 106	2 800	2 762	2 710	2 622	2 220

C. Oreochromis niloticus

Nº de transcritos originales: 3 399

Nº de lecturas artificiales (30x): 650 840

	Nº transcritos		Completitud de los transcritos				
Programa	Totales	> 500n	>80%	>85%	>90%	>95%	100%
CAP3	4 859	4 749	1 275	1 187	1 098	984	781
MIRA	4 326	3 977	2 119	2 005	1 892	1 789	1 663
VELVET	5 337	3 183	2 109	2 034	1 936	1 576	13
EULER	6 793	2 900	2 046	1 980	1 920	1 832	1 684

D. Danio rerio

Nº de transcritos originales: 5 000

Nº de lecturas artificiales (30x): 733 238

	Nº transcritos		Completitud de los transcritos				
Programa	Totales	> 500n	>80%	>85%	>90%	>95%	100%
CAP3	4 695	4 529	1 722	1 544	1 340	1 070	650
MIRA	4 807	4 426	2 980	2 707	2 400	2 064	1 575
VELVET	6 199	3 170	2 213	2 031	1 808	1 384	77
EULER	9 495	2 996	2 038	1 877	1 719	1 528	1 262

IV.1.2.2. Ensamblaje de genomas

A continuación se evaluó la capacidad de los 4 programas del apartado anterior para ensamblar lecturas artificiales genómicas, y también se añadió el programa CABOG, específico para ensamblar genomas de más de 100 Kb. La evaluación se llevó a cabo con el protocolo experimental ilustrado de la figura IV.2, muy similar al utilizado en la parte de transcriptómica, con la diferencia que la valoración de las características del ensamblaje utilizaba diferentes criterios que son el N50, N90 y la coherencia entre los contigs y la secuencia original. El N50 es un valor definido como la longitud del contig (siempre partiendo desde el contig de mayor tamaño) a partir del cual está contenido el 50% de los nucleótidos ensamblados [140], en el caso de N90 es 90%. N50 y N90 son parámetros muy utilizados para evaluar la calidad de los borradores de genomas ya que dan una idea sobre la longitud de los contigs respecto a la longitud total.

Las lecturas artificiales simples se crearon en puntos aleatorios del genoma y con longitudes aleatorias entre 300 y 500 pb para que sean lo más parecidas a las lecturas reales de 454. Las lecturas extraídas se dejaron como exactas (sin errores) para simplificar los ensamblajes y dejar como única variante la complejidad del genoma del organismo, y para permitir una comprobación precisa de los contigs resultantes. En la figura IV.3 se muestra una comparación entre la distribución de las frecuencias de las coberturas de lecturas reales (SRR097627) de *Saccharomyces cerevisiae* con una cobertura aparente de 4X mapeadas sobre el cromosoma III de esta especie y lecturas artificiales extraídas desde el cromosoma III con la misma cobertura. Según vemos, las lecturas artificiales tienen una distribución normal de frecuencias de coberturas muy parecida a aquella en las lecturas reales. Por tanto, el criterio con el que se generan las lecturas artificiales simula en buena medida a un experimento de secuenciación.

Las muestras del genoma utilizadas incluyen todo o parte del genoma de 3 organismos con longitud y grado de complejidad creciente (tabla IV.4) para ver el comportamiento de los programas de ensamblaje frente a diferentes tipos de genomas y evaluar su capacidad a ensamblar las zonas repetitivas.

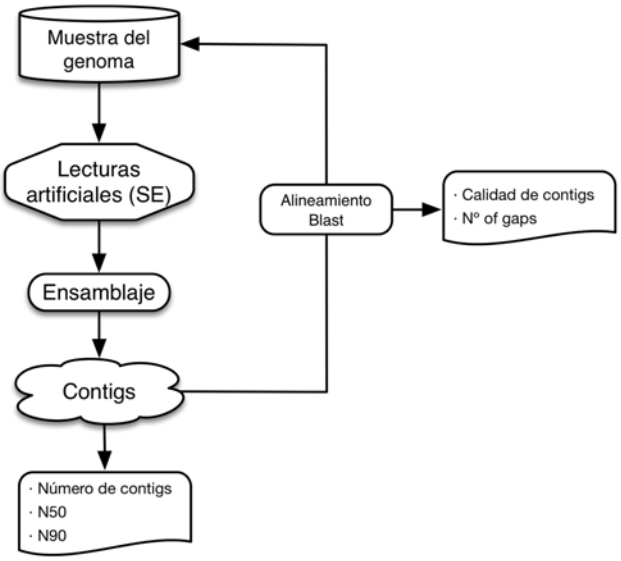

***Figura IV.2**: Las diferentes etapas de comprobación de un programa de ensamblaje de lecturas genómicas.*

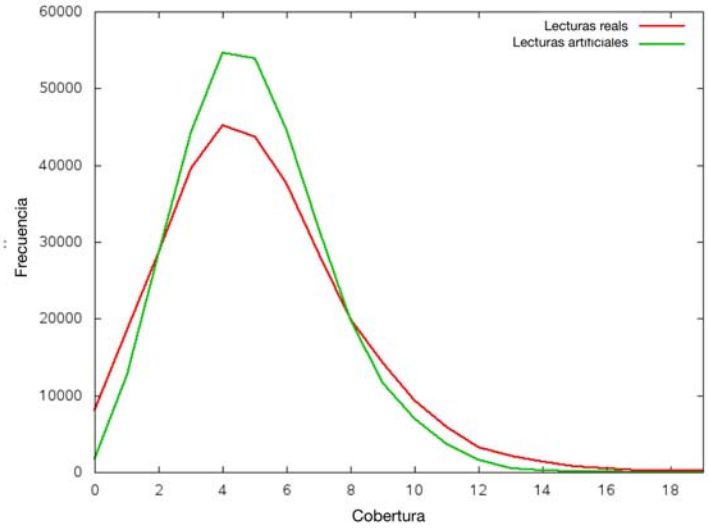

***Figura IV.3**: Distribución de la cobertura de lecturas reales y artificiales cuando la cobertura es de 4X sobre el cromosoma III de levadura*

***Tabla IV.4**: Muestras de genoma utilizadas para la evaluación de los programas*

Organismo	*Secuencia utilizada*	*Longitud*
Enterobacteria phage λ	*Todo el genoma*	*48 502 pb*
Saccharomyces cerevisiae	*Cromosoma 3*	*316 617 pb*
Homo sapiens	*Cromosoma 14 (extremo)*	*500 000 pb*

En la tabla IV.5 se recogen los resultados de contigs resultantes de los ensamblajes para las secuencias genómicas indicadas en la tabla IV.4. También se indican las características básicas de esos contigs y su calidad (basada en las identidades encontradas por Blast entre el contig y la secuencia original). Así, un contig se considera "bueno" si tiene sea un alineamiento único o varios alineamientos que están en el mismo orden en el contig y en la secuencia original. Para los contig "buenos" se incluye otro índice de calidad, que es el número de huecos (*gaps*) al ser indicadores de la discontinuidad del alineamiento entre el contig y la secuencia original; cuantos menos huecos, mejor es la reconstrucción. Se muestran solo los resultados correspondientes a la cobertura 30X por el mismo motivo que con las pruebas para transcriptómica.

Respecto al bacteriófago λ de las *Enterobacteria,* cuyo genoma es relativamente simple, los ensamblajes fueron rápidos y todos los programas produjeron un único contig cuyo tamaño estaba muy cerca del tamaño del genoma. En todos los casos el contig se alineó con la secuencia original, sin huecos (tabla IV.5-A). CAP3, MIRA y EULER produjeron el resultado más cercano al esperado.

En el caso del cromosoma III de *Saccharomyces cerevisiae* (tabla IV.5-B), cuya secuencia es más larga y que lleva más repeticiones que el bacteriófago, los programas produjeron un borrador de genoma con varios contigs, pero con cobertura alta respecto a la secuencia original (Entre 97,8% en EULER y 99,9% en MIRA). El mal resultado de EULER se debe a que generó el borrador de genoma más fragmentado (49 contigs) con alineamientos discontinuos con la secuencia original (7 huecos, mientras que los otros ensambladores no presentan ninguno). CAP3, MIRA y CABOG generaron un número de contigs más reducido con el menor número por parte de CAP3 (2 contigs). Por tanto, parece que EULER no va a ser un buen ensamblador de secuencias genómicas.

Respecto al cromosoma humano (tabla IV.5-C), aunque la cobertura de lecturas fue la misma que en los casos anteriores, los programas se mostraron más conservadores y produjeron más contigs en general. CAP3 generó menos contigs que los demás programas, pero solo 8 de sus 9 contigs fueron "buenos", y uno de los "buenos" tenía un hueco. En la figura IV.4 se ilustra la estructura del Contig9 generado por CAP3 en comparación con la secuencia original. El orden invertido de los alineamientos entre el contig y la secuencia original y los errores que se observan en las lecturas ensambladas indican que los extremos 3' y 5' de esta zona del genoma comparten una secuencia repetitiva que hizo equivocarse al ensamblador. En el caso de VELVET y EULER el borrador de genoma generado fue mucho más fragmentado que los demás (368 y 2 564 contigs respectivamente). Solo 94 (3,7%) de los contigs de EULER fueron superiores a 500 pb. De estos 94 contigs, 2 fueron malos y los demás, a pesar de ser "buenos", incluían 20 huecos, mientras que VELVET generó 368 contigs, 35 (9,5%) de ellos superiores a 500 pb, y todos fueron "buenos" y sin huecos. MIRA tuvo claramente el mejor resultado con menor número de contigs, y mayor N50 (298 365), además todos sus contigs fueron "buenos", carecían de huecos y cubrían toda la secuencia del genoma original (100,4%). CABOG tuvo un resultado parecido al de MIRA, aunque con menor N50 (108 539) y menor cobertura de la secuencia original (94,2%). Por tanto, parece

que MIRA y CABOG son los ensambladores más idóneos para reconstruir secuencias genómicas de alta complejidad.

***Tabla IV.5**: Resultados de la evaluación de los programas de ensamblaje de novo de secuencias genómicas con una cobertura 30X*

(A) Enterobacteria phage λ
Parte del genoma: Todo
Nº de lecturas artificiales (30X): 3 643

	Tiempo de ejecución (s)	Contigs				Contigs "buenos"			
		Nº	> 500 nt	N50	N90	Nº	Suma (nt)	Cob. (%)*	Huecos
CAP3	73	1	1	48 415	-	1	48 415	99,8	
MIRA	75	1	1	48 415		1	48 415	99,8	0
CABOG	78	1	1	48 347	-	1	48 347	99,7	0
VELVET	39	1	1	48 409	-	1	48 409	99,8	0
EULER	10	1	1	48 415	-	1	48 415	99,8	0

(B) Saccharomyces cerevisiae
Parte del genoma: Cromosoma 3
Nº de lecturas artificiales (30X): 23 789

	Tiempo de ejecución (s)	Contigs				Contigs "buenos"			
		Nº	> 500 nt	N50	N90	Nº	Suma (nt)	Cob. (%)*	Huecos
CAP3	640	2	2	302 907	11 907	2	314 814	99,4	0
MIRA	496	5	5	186 396	23 565	5	316 298	99,9	0
CABOG	528	4	4	185 448	23 205	4	313 165	98,9	0
VELVET	300	21	6	114 172	70 111	6	312 719	98,8	0
EULER	82	49	9	69 034	21 783	9	309 615	97,8	7

(C) Homo sapiens
Parte del genoma: últimos 500kb del cromosoma 14
Nº de lecturas artificiales (30X): 37 581

	Tiempo de ejecución (s)	Contigs				Contigs "buenos"			
		Nº	> 500 nt	N50	N90	Nº	Suma (nt)	Cov. (%)*	Huecos
CAP3	1 586	9	9	149 272	33 754	8	416 534	83,3	1
MIRA	1 430	7	7	298 365	193 585	7	502 085	100,4	0
CABOG	677	13	13	108 539	29 820	13	470 800	94,2	0
VELVET	596	368	35	49 352	1 554	35	468 406	93,7	0
EULER	152	2 564	94	1 642	72	92	275 654	55,1	20

(*) Porcentaje del genoma original cubierto por los contigs ensamblados

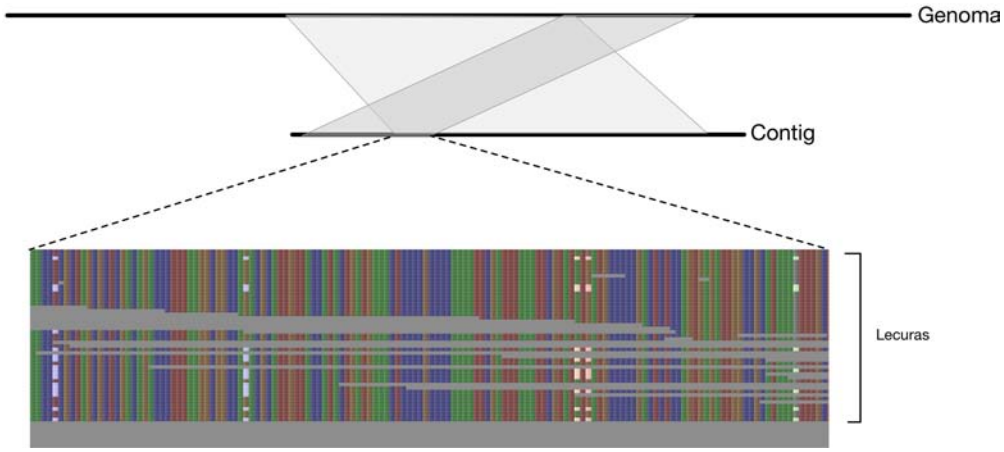

Figura IV.4: *Ejemplo de error de ensamblaje detectado en el Contig9 del ensamblaje CAP3 para el fragmento del cromosoma 14 del genoma humano*

Para comprobar si el comportamiento de los ensambladores podría ser dependiente de la cobertura, analizamos el número de huecos que presentaban los contigs "buenos" producidos en el caso del genoma humano (tabla IV.5-C) para coberturas que van de 5X a 60X (tabla IV.6). En el caso de CAP3, MIRA y EULER se nota un aumento de huecos cuando la cobertura era baja, aunque EULER siempre produjo huecos en todas las condiciones analizadas.

Tabla IV.6: *Número de huecos en los contigs "buenos" según la cobertura en el caso del fragmento de cromosoma 14 humano*

	Cobertura						
	5x	***10x***	***20x***	***30x***	***40x***	***50x***	***60x***
CAP3	*4*	*1*	*3*	*1*	*2*	*0*	*1*
MIRA	*8*	*1*	*1*	*0*	*0*	*0*	*0*
CABOG	*0*	*0*	*0*	*0*	*0*	*0*	*0*
VELVET	*1*	*1*	*0*	*0*	*0*	*0*	*0*
EULER	*13*	*25*	*28*	*20*	*12*	*13*	*9*

En conclusión, los programas de ensamblaje estudiados se mostraron eficientes para ensamblar genomas simples, mientras que en genomas más complejos se ponen de manifiesto mejor sus diferencias. Entre ellos destacan MIRA y CABOG al producir el mejor resultado en cuanto a número de contigs y su calidad. Como recomendación, conviene utilizar MIRA con una cobertura ≥30X. Para ensamblajes con poca cobertura o que ésta sea poco homogénea, es más recomendable utilizar CABOG ya que genera contigs más fiables en estas condiciones.

IV.1.2.3. Combinación de ensamblajes de varios programas

Puesto que no hay programa bioinformático óptimo para ningún problema biológico, un buen criterio consiste en combinar los resultados de distintos algoritmos con los mismos datos. En el caso de los ensamblajes, parece conveniente combinar los contigs de dos o más programas de ensamblaje para obtener el mejor ensamblaje posible.

Ensamblajes de transcriptoma

En el apartado IV.1.2.1 hemos visto que los programas MIRA (algoritmo de tipo OLC) y EULER (algoritmo de De Bruijn) generaron los mejores transcriptomas. Así, suponiendo que hay lecturas que se ensamblarán mejor con un algoritmo que con otro, definimos un método para reconciliar los contigs de estos dos ensambladores para sacar un solo transcriptoma. Para este propósito utilizamos CAP3, un programa inicialmente destinado a ensamblar lecturas de tipo Sanger y apto para ensamblar secuencias largas como los contigs resultantes de un ensamblaje de transcritos. Como resultado tendremos *supercontigs* donde habrá contigs de MIRA y EULER, y singulones que son contigs que no encontraron con quién alinearse. La combinación de estos dos grupos de secuencias esperamos que devuelva un transcriptoma final mejor que el de cualquiera de los ensambladores por separado (figura IV.5).

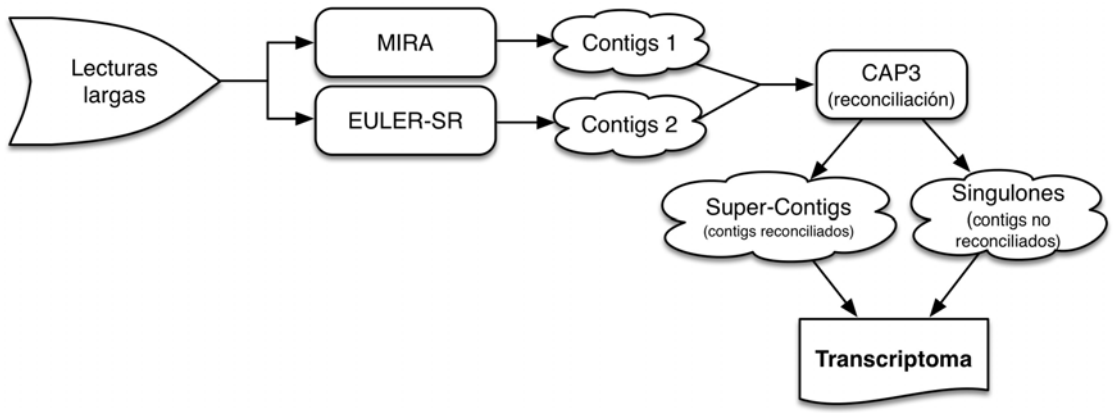

***Figura IV.5**: Reconciliación entre los contigs de MIRA y EULER utilizando CAP3*

La validación de esta estrategia fue realizada en nuestro laboratorio por Noé Fernández Pozo [87] para ensamblar el transcriptoma de *Pinus pinaster* y presentarlo a la comunidad científica en la base de datos EuroPineDB [141]. En la tabla IV.7 se observa cómo el ensamblaje en supercontigs con CAP3 devuelve un número de singulones más supercontigs menor que la suma de contigs de EULER y MIRA por separado (primera fila de la tabla), y hay mayor número y porcentaje de transcritos completos y únicos. También se ha validado posteriormente en nuestro laboratorio con otros transcriptomas, tanto de especies animales como vegetales [140, 142, 143].

***Tabla IV.7**: Comparación del ensamblaje de lecturas de 454/Roche para generar el transcriptoma de Pinus pinaster. La tabla está sacada del manuscrito en preparación del programa Full-LengtherNext (Seoane et al, en preparación)*

	Euler-SR		MIRA3		CAP3[1]	
	#seqs	%	#seqs	%	#seqs	%
Unigenes	24147	100%	50813	100%	41246	100%
Unigenes > 500 bp	11729	48.57%	18453	36.32%	20115	48.77%
Longest unigene (bp)	4483		7450		7450	
With ortologue[2]	16779	69.49%	35334	69.54%	28314	68.65%
Different ortologue IDs	9997	59.58%	14334	40.57%	13452	47.51%
Complete transcripts	2766	16.49%	6328	17.91%	6164	21.77%
Different complete transcripts	2606	15.53%	4055	11.48%	4640	16.39%
Misassembled	11	0.07%	139	0.39%	30	0.11%
Without ortologue[2]	7362	30.49%	15479	30.46%	12932	31.35%
Coding	785	10.66%	2191	14.15%	1815	14.03%
Putative coding	492	6.68%	1772	11.45%	1490	11.52%
Putative ncRNA	6	0.08%	5	0.03%	8	0.06%
Unknown	6085	82.65%	11511	74.37%	9619	74.38%
Mapped reads[3]	443 976	59.50%	637321	85.41%	651 643	87.33%

[1] Due to its overlap-layout-consensus design for Sanger sequencing, CAP3 cannot be used with the huge amount of reads provided by any NGS method. It has been therefore used for reconciliation of the assemblies obtained from Euler-SR and MIRA3.
[2] Percents for subclassifications of this category were calculated using this line as 100% reference.
[3] Mapping was performed with Bowtie 2.0 using the default parameters (Langmead *et al.*, 2009) and the useful reads as input.

Los programas CAP3 y MIRA utilizan el formato ACE para incluir a los datos de las lecturas que han permitido formar la secuencia consenso del contig, mientras que EULER no utiliza este formato. Pensando en que el ensamblaje en formato ACE podría servir para inspeccionar la calidad de ensamblaje de un transcrito (véase más adelante, apartado IV.1.3), y que también podría servir para detectar SNP con GigaBayes [131], decidimos crear un ACE de los supercontigs generados por CAP3, como se ilustra en la figura IV.6. Para ello, se desarrolló un script en Ruby (véase materiales y métodos; apartado III.3.5.2) que importa la información de los dos ficheros ACE y en el fichero ACE del CAP3 reemplaza los contigs del MIRA con las lecturas ensambladas que les corresponden (figura IV.6).

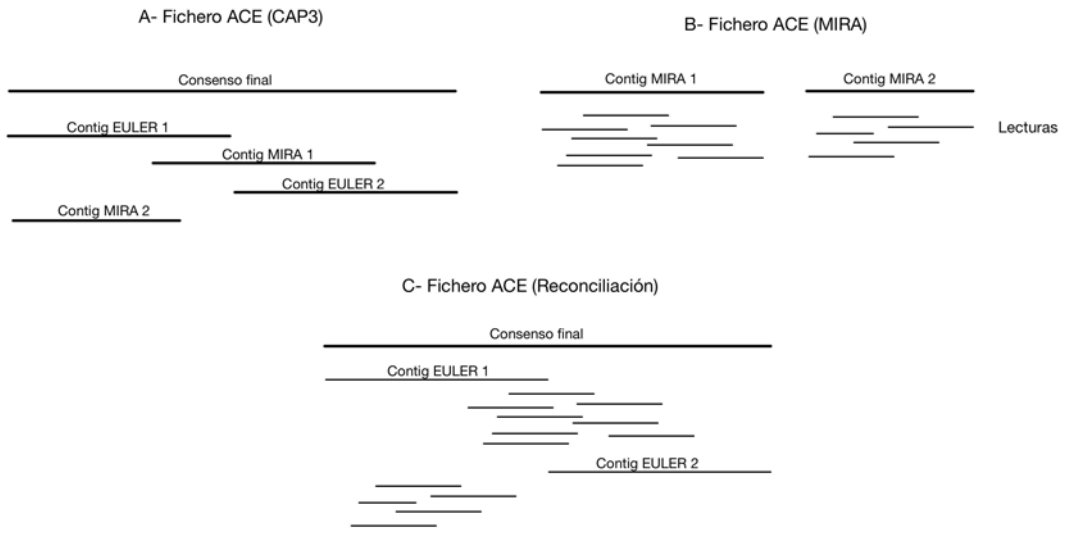

***Figura IV.6:** Combinación de los ficheros ACE de MIRA y CAP3. Los contigs de MIRA (A) se reemplazan con las lecturas que les corresponden (B) para generar un nuevo fichero ACE (C)*

Ensamblajes de genoma

En los ensamblajes de genoma, no es tan fácil hacer una fusión de ensamblajes, ya que puede causar un borrador de genoma con duplicaciones. Además, tiene que hacerse con un programa diferente, GAM-NGS [118], en el que hay que definir un ensamblaje de referencia o maestro, que luego se puede extender con otro ensamblaje que denominan "esclavo". Comparativemente a otros programas probados como GARM [144], GAM-NGS tiene la ventaja de utilizar la información de mapeo las secuencias de lecturas sobre los contigs o scaffolds para definir regiones llamadas "bloques" que representan el mismo locus entre los diferentes contigs o *scaffods*, lo cual permite una fusión más fiable de estos. Dado que cada vez se secuencian menos genomas con 454/Roche y más con secuencias cortas de Illumina (cuyos ensambladores incluyen diferentes versiones de algoritmos basados en los grafos de De Bruijn), otra manera de combinar distintos ensamblajes es usar diferentes *k*-meros y combinarlos. Algunos ensambladores, como SOAPdenovo [66] lo hacen por sí solos, mientras que otros, como Trinity, solo son capaces de usar un *k*-mer. En los próximos capítulos veremos con más detalle la estrategia seguida para ensamblar el genoma de una bacteria con 454/Roche, y el del lenguado combinando 454/Roche e Illumina.

IV.1.2.4. Tamaño real del inserto en las lecturas pareadas

Para poder ordenar los contigs obtenidos se utilizan lecturas pareadas, con las que definimos los *scaffolds*. El tamaño de inserto en las lecturas pareadas (PE o MP) varía de una librería a otra según las plataformas de secuenciación y el protocolo experimental utilizado. La mayoría de los programas de ensamblaje y de *scaffolding* requieren incluir la media y la desviación típica del tamaño real de los insertos. Para obtener este valor, desarrollamos un script en Ruby (véase materiales y métodos; apartado III.3.5.3) con el que ensamblamos lecturas pareadas que sigue el protocolo ilustrado en la figura IV.7. Las secuencias pareadas usadas para el ensamblaje se mapean con Bowtie sobre los contigs resultantes para medir a qué distancia caen los extremos de las secuencias pareadas y así estimar el tamaño medio y la desviación típica.

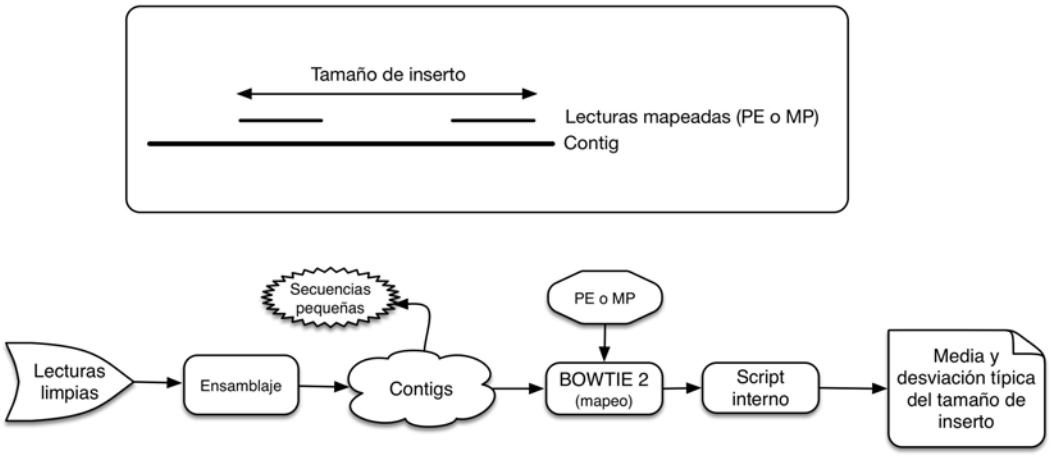

***Figura IV.7**: Protocolo utilizado para calcular la media y la desviación típica del tamaño de inserto en lecturas pareadas. El programa usado para el ensamblaje es, en cada caso, el que requiere los valores de tamaño de inserto, como por ejemplo, CABOG*

IV.1.3. CoMiner: validación de ensamblajes

En cualquier tipo de ensamblaje *de novo* existen errores. La identificación de los ensamblajes erróneos constituye una tarea difícil debido a la gran cantidad de datos y a la variación de la calidad de estos datos por culpa de complicaciones mecánicas y bioquímicas de los secuenciadores. Para la reconstrucción de un transcriptoma o genoma fiable, la corrección de errores en los ensamblajes requiere esfuerzos adicionales de validación manual [145]. La manera habitual de medir la calidad de los ensamblajes se basa en estadísticas simples, como el contig más largo, la longitud media de los contig, la suma de las longitudes y el N50. De hecho, se suele dar preferencia a los contigs largos [146], sin reparar muchas veces en que puedan ser quiméricos o resultantes de un ensamblaje erróneo.

Se han desarrollado validadores de ensamblajes basados en parámetros más complejos, como es Hawkeye [147]. Sirve para facilitar la inspección visual de los ensamblajes en el que se muestran los puntos de conflicto para reducir al mínimo el tiempo dedicado a la identificación, edición y corrección. A diferencia de otros editores de contigs como GAP5 [148], Hawkeye combina predictores computacionales con visualización interactiva. Su principal problema está en que es manual, por lo que pierde utilidad a medida que aumenta la cantidad de contigs de un ensamblaje.

Por eso, aunque la inspección visual y edición manual sean métodos eficaces, estos pueden constituir unas tareas engorrosas, particularmente para ensamblajes con los datos de secuenciación de nueva generación (NGS), por esta razón se desarrolló un programa de validación automática de los contigs basado en criterios independes, llamado *amosvalidate* [146]. No obstante, esta herramienta solo marcaba las regiones que aparecían mal ensambladas, y la corrección requiere una nueva visualización y una edición manual utilizando *Hawkeye*.

En este trabajo nos planteamos la detección de errores de ensamblaje y su corrección automática, para lo que desarrollamos un programa llamado CoMiner (verse el apartado III.3.5.8 de materiales y métodos) que fue programado en Ruby y comprobado en un iMac dual core a 3,06 GHz con 4 Gb de RAM. El programa lee y escribe ensamblajes en el formato ACE [149], que es un formato generado por varios programas basados en el algoritmo OLC como los tradicionales Phrap, CAP3 y GAP4-5, y los desarrollados para la NGS de lecturas largas (Newbler, CABOG, Arachne, Minimus y TIGR Assembler).

Con el objetivo de incrementar la fiabilidad del ensamblaje sin la intervención humana, el algoritmo de CoMiner se divide en 4 etapas principales (figura IV.8): (i) descubrimiento de las regiones de alta entropía (HER, por *High-entropy region*); (ii) identificación de las lecturas conflictivas; (iii) edición de las lecturas (recorte o eliminación); (iv) verificación del contig y escritura del nuevo fichero ACE.

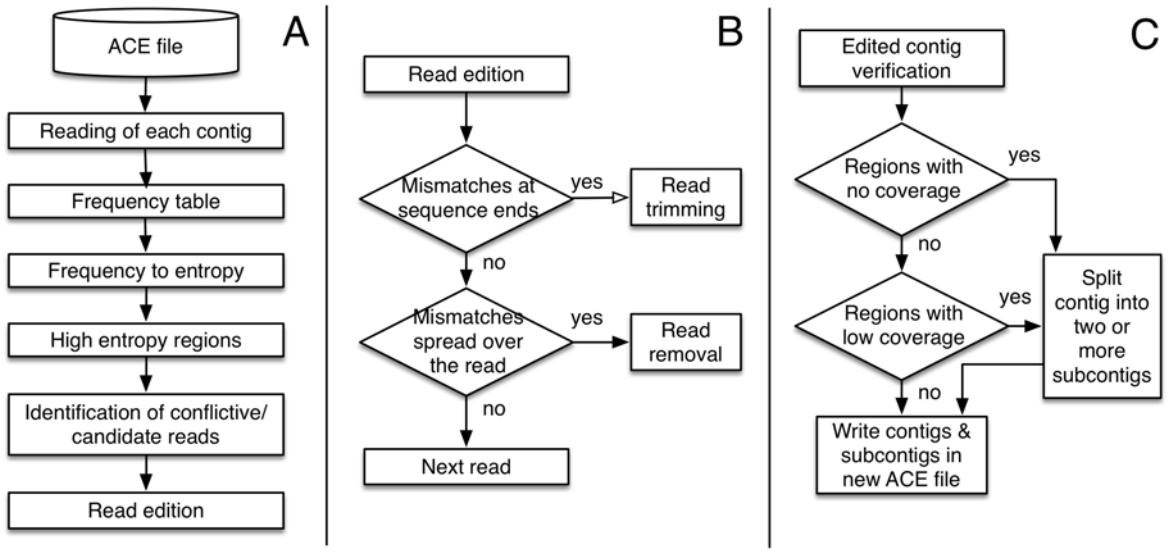

Figura IV.8*: (tomada de [150]) Flujo de trabajo del algoritmo de CoMiner. A. Descubrimiento de las regiones de alta entropía e identificación de las lecturas conflictivas. B: Edición o eliminación de las lecturas conflictivas en un contig según la distribución de errores. C: Una vez editadas las lecturas conflictivas, se verifica la cobertura del contig. Este se divide en dos o más contigs si es necesario, luego se guarda en el fichero ACE*

Veamos cada una de estas etapas con más detalle:

(i) Descubrimiento de HER: El objetivo de esta etapa es localizar los fragmentos donde las lecturas no alinean perfectamente. Se eligió una medida para definir la calidad de un alineamiento, la entropía de la secuencia consenso en la posición i [-H(i)] tal como se describió en [151]. Se calcula entonces la frecuencia de cada uno de los 4 nucleótidos en cada posición de la secuencia consenso, lo que se utiliza para calcular la entropía en cada posición del consenso. Los SNP y discordancias

(*mismatch*) se consideran equivalentes y no afectan significativamente a los ensamblajes. A los datos por posición se les aplicó una transformada rápida de Fourier como se describe en [151] para convertir los picos en regiones de alta entropía. Los rangos de la secuencia donde la transformada de Fourier de la entropía es superior a un valor de corte que corresponde a la entropía mediana se considerarán los HER objeto de análisis posterior.

(ii) Identificación de las lecturas conflictivas: Se realizan varios cálculos para saber si un HER se debe a una o más lecturas mal ensambladas o que fue causado por discordancias distribuidas entre todas las lecturas implicadas. Para distinguir entre estas posibilidades, se calcula la frecuencia de la discordancia de una lectura r [$F_{read}(r)$] de la siguiente manera: para un contig que contiene un número de lecturas m en el cual se definió un número k de HER, donde n(j) es la longitud de un HER, $F_{read}(r)$ se calcula dividiendo el número total de discordancias de una lectura r dentro de todos los HER presentes en el consenso entre el número total de los nucleótidos implicados en todos los HER:

$$F_{read}(r) = \sum_{j=1}^{k}\sum_{i=1}^{n} err(i,j) \Big/ \sum_{j=1}^{k} n(j)$$

La frecuencia total de discordancias del contig (F_{contig}) se calcula dividiendo el número total de los discordancias de todos las lecturas del contig dentro de todos los HER entre la longitud total de todas las regiones HER en cada lectura, de la siguiente manera:

$$F_{contig} = \sum_{r=1}^{m}\sum_{j=1}^{k}\sum_{i=1}^{n} err(i,j,r) \Big/ \sum_{r=1}^{m}\sum_{j=1}^{k} n(j,r)$$

Los dos valores Fread y Fcontig definen un valor robusto Fcutoff:

$$F_{cutoff} = F_{contig} + K \times \left(\sum_{r=1}^{m} \left| F_{read}(r) - F_{contig} \right| \Big/ m \right)$$

Donde K = 1,4826. Por tanto, las lecturas con $F_{read}(r) > F_{cutoff,}$ se consideran como conflictivas y candidatas para la edición.

(iii) Edición de las lecturas candidatas: Se analiza la distribución de las discordancias en las lecturas candidatas como muestra en la figura IV.9, orientando el recorte o la eliminación de las lecturas conflictivas en función de la distribución de las discordancias.

(iv) Verificación y escritura del contig: La edición de las lecturas puede modificar la cobertura del contig, lo que hace que algunas regiones puedan quedarse sin cobertura. El algoritmo busca este tipo de situaciones y divide el contig en dos subcontigs nuevos, cada uno con una secuencia consenso nueva e independiente.

Hay casos especiales donde el contig incluye una zona de dos lecturas solapantes flanqueadas por una cobertura de una sola lectura (figura IV.10). Este contig se dividirá en dos subcontigs independientes cuando el fragmento solapante es inferior a 40 nt o la identidad es inferior a 90%. Los contigs sin editar, contigs editados y nuevos subcontigs se guardarán en un fichero ACE nuevo.

***Figura IV.9**: (tomada de [150]) Diferentes instancias de la distribución de las discordancias en una lectura conflictiva. A: todas las discordancias localizadas en un solo extremo; entonces, los nucleótidos en la distancia d se recortan desde la lectura siempre que d < 40% de la longitud de la lectura y que la lectura restante es superior a 40 nt. B: Existen discordancias en ambos extremos; de nuevo los nucleótidos en las distancias d1 y d2 se recortan siempre que (d1 + d2) < 40% de la longitud de la lectura y que la longitud de la lectura restante es superior a 40 nt. C: Las discordancias están repartidas por toda la lectura; cuando el número de discordancias es superior a 2% de la longitud de la lectura, se elimina toda la lectura*

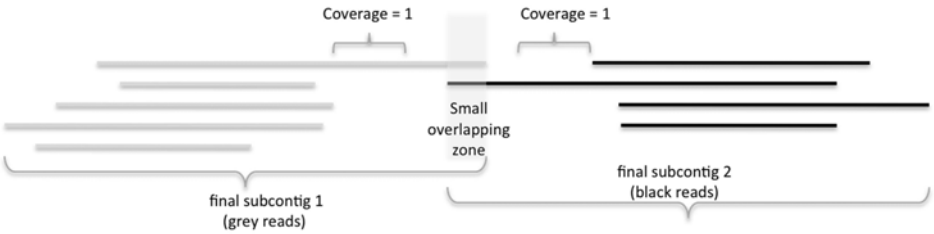

***Figura IV.10**: (tomada de [150]) Ejemplo de un contig después de la edición de lecturas, donde solo dos lecturas conectan ligeramente dos posibles subcontigs. CoMiner divide éste contig en dos nuevos subcontigs*

Las capacidades de CoMiner se analizaron en ensamblajes de transcriptoma y genoma (tabla IV.8). Un total de 1 110 923 lecturas 454/FLX del transcriptoma de *Solea senegalensis* (lenguado) se preprocesaron con SeqTrimNext (`http://www.scbi.uma.es/seqtrimnext`) [152] y luego se ensamblaron con MIRA3 (`http://sourceforge.net/projects/mira-assembler`) [139] con los parámetros estándares para datos 454/FLX. En el ensamblaje resultante (tabla IV.8 columna Transcriptome), CoMiner detectó HER en 33 912 contigs (44,1%), pero solo editó 21 002 (27,3%), que eran contigs donde al menos un HER fue causado por discordancias concentradas en una lectura al menos. En la figura IV.11 se muestra un ejemplo donde el algoritmo identificó una lectura que incluye varias discordancias que fueron responsables del ancho HER central, Se recortó la parte 3' de esta lectura, al incluir todas las discordancias. Después de la edición automática realizada por el CoMiner, un nuevo análisis de entropía

mostró que el HER despareció, lo que indica que el nuevo contig era más fiable que el contig inicial.

La integridad de la mayoría de los contigs no fue afectada por la edición de CoMiner, pero 146 (0,7%) de estos se dividieron en dos subcontigs y otros 2 contigs se dividieron en 3 diferentes subcontigs cada uno. Hay que considerar que cuando un subcontig consiste en una sola lectura, éste no cuenta como contig y la lectura se elimina del conteo final de contigs. Un ejemplo de la división de contig se muestra en la figura IV.12, donde un contig de 1570 pb se dividió en dos subcontigs más pequeños. Otro ejemplo de esta situación puede ser el contig quimérico group1_solea_c8241 de 1500 nt, ya que se dividió en un subcontig 5' de 887 nt con similitud a "ORM1 like protein" (B0V340; $E=10^{-107}$) de 153 amino acidos, y un subcontig 3' de 601 nt sin ninguna similitud en las bases de datos. Como resultado, la edición automática de 27,3% de los contigs no aumentó significativamente el número de contigs y no cambió significativamente los parámetros generales del ensamblaje (tabla IV.8, columna Transcriptome), mientras que la fiabilidad de los contigs resultó presumiblemente mejorada.

***Tabla IV.8**: (tomada de [150]): Resultados de dos ensamblajes antes (-) y después (+) del tratamiento CoMiner*

	Transcriptome		Genome	
CoMiner	–	+	–	+
Contig #	76 824	76 894	35 777	35 823
Mean contig size (nt)	429	428	471	470
N50 (nt)	484	482	506	505
N90 (nt)	251	252	307	307
Edited contigs		21 002		1465
Split contigs		146		74
Mapped contigs			35 412	35 458
Mapped nt			3 362 691	3 362 923

El ADN ensamblado del genoma fue analizado con 317 692 lecturas genómicas de *Arabidopsis thaliana* (SRX105465). Estas fueron preprocesadas con SeqTrimNext y ensambladas con CAP3 [105] (`http://seq.cs.iastate.edu/`) para obtener 35 777 contigs (tabla IV.8, columna Genome). CoMiner detectó 3 259 contigs (9,1%) con uno o más HERs, pero solo editó 1465 (4,1%), 74 de ellos fueron divididos en dos o más contigs. Los contigs antes y después del tratamiento se analizaron por comparación con el genoma de *A. thaliana* con el mismo algoritmo utilizado en el apartado IV.1.2.2 para comprobar un posible aumento de la fiabilidad de los contigs. Se mapeó un total de 35 412 (98,97%) y 35 458 (98,98%) contigs respectivamente, proporcionando un total de 3 362 691 y 3 362 923 nucleótidos mapeados respectivamente (tabla IV.8, columna Genome). Cuando el mapeo se realizó con un programa de mapeo más restrictivo como Bowtie2 [153], se mapearon 8453 contigs originales (23,62%) y 8494 contigs editados por CoMiner (23,71%). La edición incrementó ligeramente el número de contigs mapeados y de los nucleótidos. Por desgracia, el incremento está en el mismo rango que el incremento de número total de contigs, por lo que se requieren más análisis para proporcionar una significación estadística a este incremento débil.

Como hemos visto, CoMiner fue testeado con datos de transcriptoma y genoma y la calidad de los contigs editados según parece fue mejorada. Sin embargo, habría que analizar más ensamblajes reales en trabajos futuros para proporcionar una significación estadística a los resultados presentados en este trabajo. Los resultados de CoMiner siempre pueden ser analizados y visualizados para buscar más errores utilizando amosvalidate y Hawkeye, con el mismo objetivo de disminuir los esfuerzos dedicados a la edición manual.

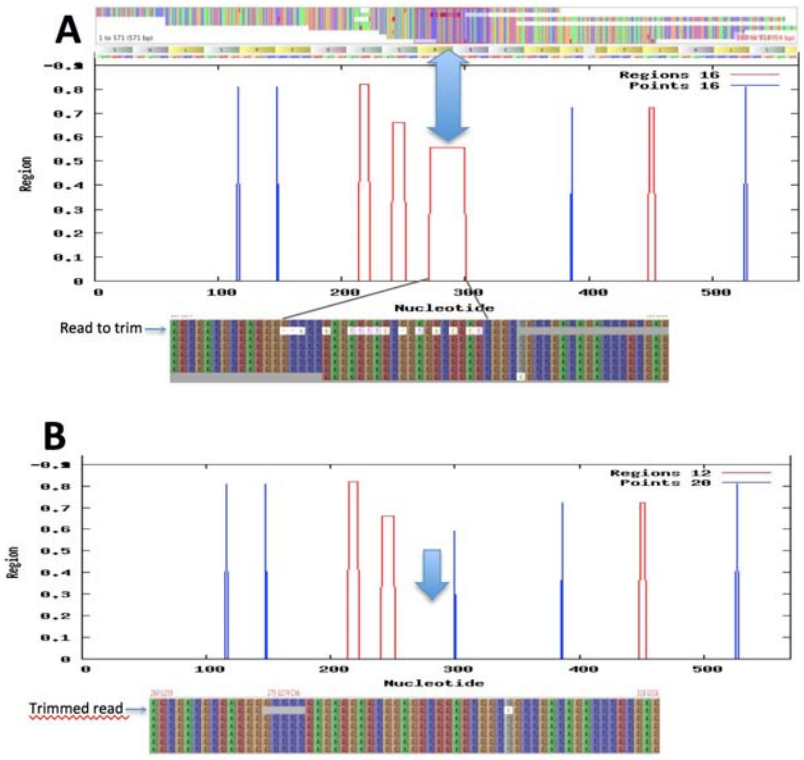

***Figura IV.11**: (tomada de [150]) Ejemplo de un contig de 571 pb con varios HER antes (A) y después (B) de la edición automática de CoMiner. Después del análisis de entropía, los HER que incluyen un solo nucleótido se consideran errores puntuales (en azul), mientras que los HER reales (en rojo) abarcan dos o más nucleótidos. El HER (nt 273-296) está marcado con una doble-flecha que muestra que todas las discordancias provienen de una sola lectura. Después de la edición (B), este ancho HER desapareció. Los otros tres HER más pequeños seguían presentes, lo que indica que sus discordancias no se concentraban en una sola lectura, y que por eso no se editaron*

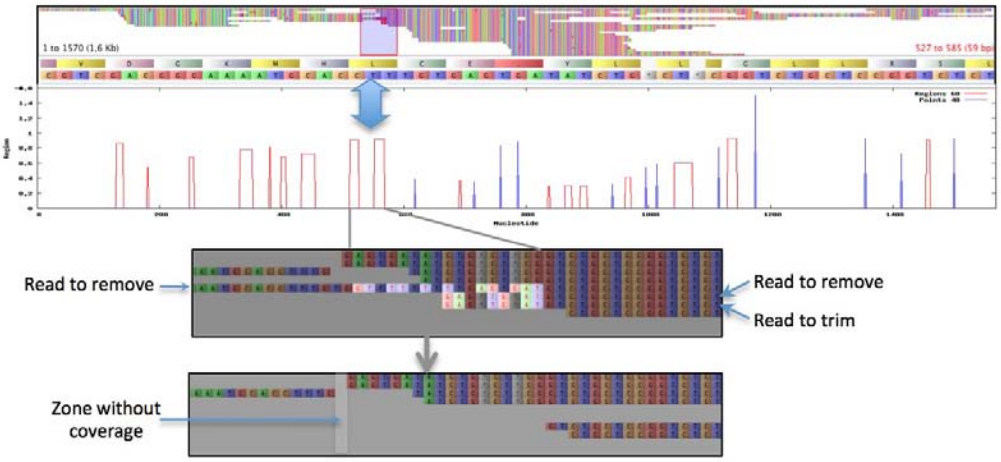

***Figura IV.12**: (tomada de [150]) Ejemplo de un contig de 1570 pb con varios HER reales en rojo. La flecha señala un par de HER que fueron resueltos recortando el lado izquierdo de una lectura y eliminando dos otras lecturas. La edición causó un gap en el contig que el CoMiner resolvió dividiendo el contig en dos subcontigs, el subcontig izquierdo a 551 nt y el subcontig derecho a 1018 nt*

IV.1.4. Ordenamiento de los scaffolds y acabado de los genomas

Una de las fases importantes en los ensamblajes genómicos, es la fase del ordenamiento de los contigs o *scaffolds* en los posibles cromosomas. La utilización a las nuevas tecnologías de secuenciación cada vez más accesibles implica un incremento continuo las secuencias genómicas de especies disponibles. Esta información, se puede aprovechar para guiar el acabado de los genomas estudiados.

Para ordenar y orientar los contigs o *scaffolds* de un borrador del genoma, desarrollamos un programa en perl llamado ICMapper (Instant Contig Mapper) (ver materiel y métodos; apartado III.3.5.9). Este programa cuyo algoritmo estaba inspirado en el algoritmo utilizado por el programa Oslay [154], aprovecha la similitud y la sintenia entre el genoma de la especie estudiada y el genoma de una especia de referencia. Se basa esencialmente en los alineamientos que devuelve Blast entre el borrador del genoma y el genoma acabado de referencia para formar un alineamiento largo representativo por cada contig o scaffold.

Los resultados dependerán de la elección del genoma utilizado como referencia. De hecho, para tener buenos resultados, el genoma de referencia y el borrador deben tener una buena sintenia, de ahí que deba ser una especie estrechamente relacionada con la nuestra para que tenga regiones comunes o parecidas que formen los alineamientos representativos de los contigs o *scaffolds* y establezcan el orden correcto. En la web de NCBI, existe una gran gama de genomas libres para la descarga (`http://www.ncbi.nlm.nih.gov/genome/browse/`)

El flujo de trabajo de ICMapper está ilustrado en la figura IV.13. Las diferentes etapas de procesamiento del programa son las siguientes:

(i) Realizar un Blast entre el borrador de genoma y el genoma de referencia. Hay que desactivar la opción de filtración de las regiones de baja complejidad en el ADN para alinear todas las zonas de genoma y evitar huecos en ellas.

(ii) Filtrar los HSP por tamaño (el usuario definirá un tamaño mínimo [por defecto 100 nt])

(iii) Agrupar los HSP primero por separación entre las diagonales (por defecto es de 7000 nt) y segundo por longitud de los huecos (por defecto 10 000 nt), con lo que se formarán clústeres de HSP (figura IV.14-A).

(iv) Convertir los clústeres de HSP en alineamientos resumidos que consisten en un alineamiento extendido con huecos en medio (figura IV.14-B).

(v) Tratamiento de las repeticiones en los alineamientos resumidos (figura IV.14-B). Un alineamiento se considera repetido cuando una misma zona de la secuencia del borrador de genoma que se alinea con dos zonas distintas del genoma de referencia. Cuando se trata de repeticiones completas, se descartan todos los alineamientos resumidos repetidos. En el caso de una repetición parcial, se conserva el alineamiento resumido más largo y se descartan los demás.

(vi) Definir el alineamiento resumido más largo como el alineamiento representativo del contig o scaffold.

(vii) Ordenar las secuencias del borrador de genoma en función de las posiciones de inicio del alineamiento resumido en el genoma de referencia (figura IV.14-C). Se les afecta también una orientación de acuerdo con el genoma de referencia.

N.B. Las etapas 2 a 6 se repiten por cada secuencia del borrador de genoma.

ICMapper fue probado en un PC Core 2 Duo E6750 y 2 GB de RAM. En la figura IV.15 se muestra un ejemplo de una representación Dot-plots de los alineamientos entre dos diferentes especies de bacterias del mismo género. Los alineamientos brutos (figura IV.15-A) aparecen muy pequeños y dispersos, mientras que los alineamientos representativos generados por ICMapper (figura IV.15-B) aparecen más largos y ocupando la diagonal que ilustra la buena sintenia entre esas dos bacterias y por lo cual un ordenamiento fiable del borrador de genoma.

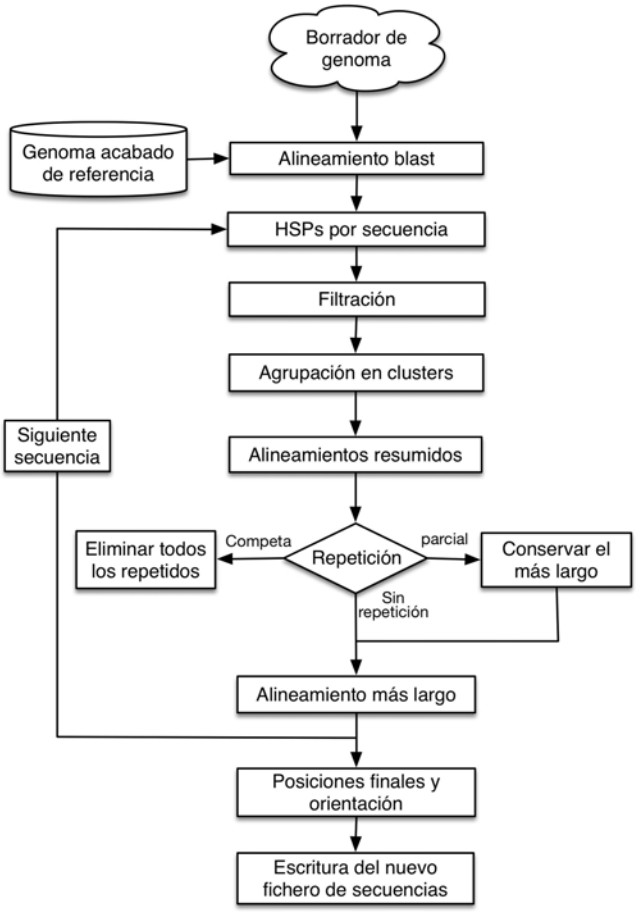

***Figura IV.13**: Esquema del flujo de trabajo del programa ICMapper*

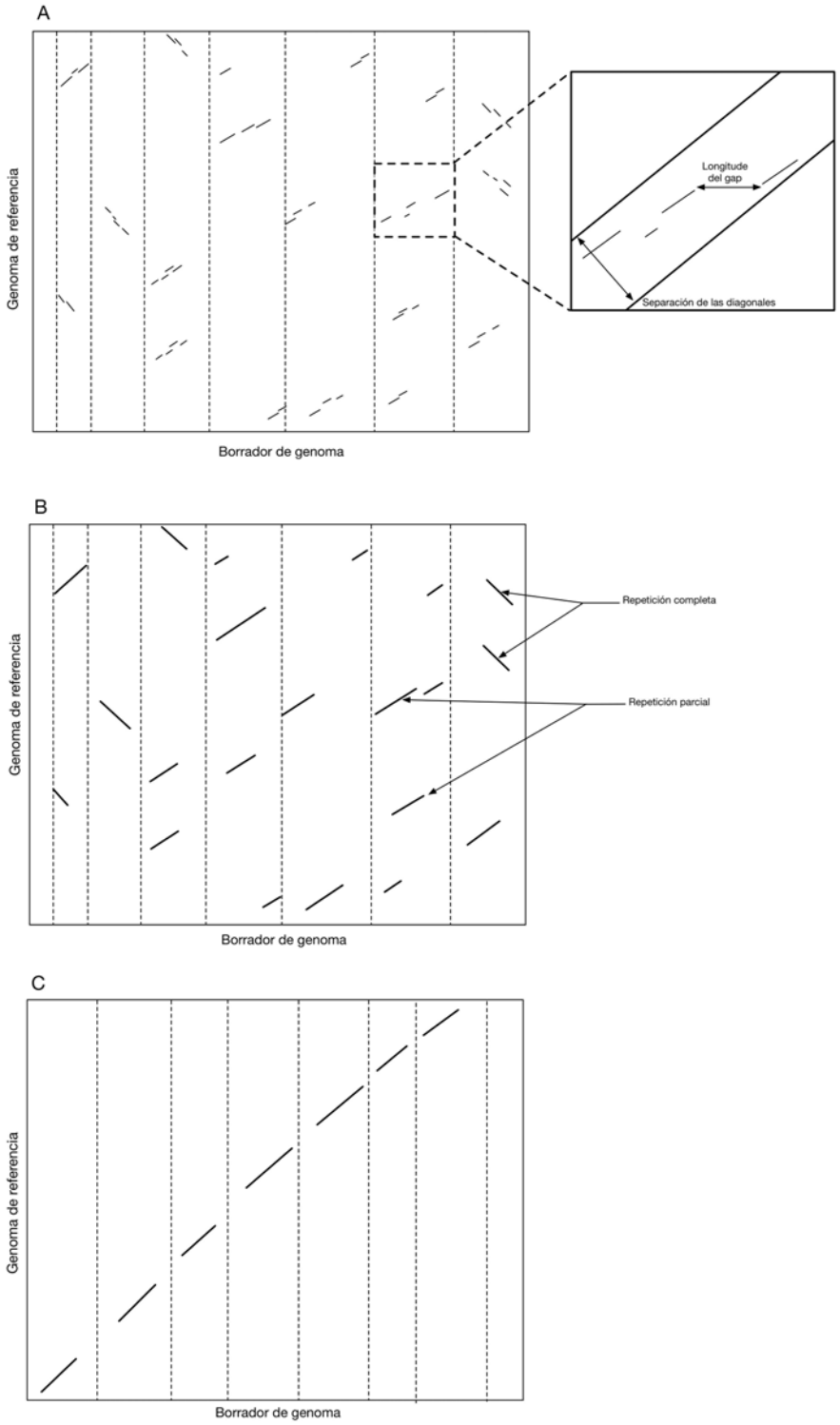

Figura IV.14*: Esquema de las diferentes etapas del tratamiento de los HSPs de Blast con ICMapper. A: clusterización de los HSP por diagonal y por longitud de huecos (gaps). B: Formación de alineamientos resumidos por cada clúster. C: Eliminación de las repeticiones y ordenamiento de las secuencias del borrador de genoma*

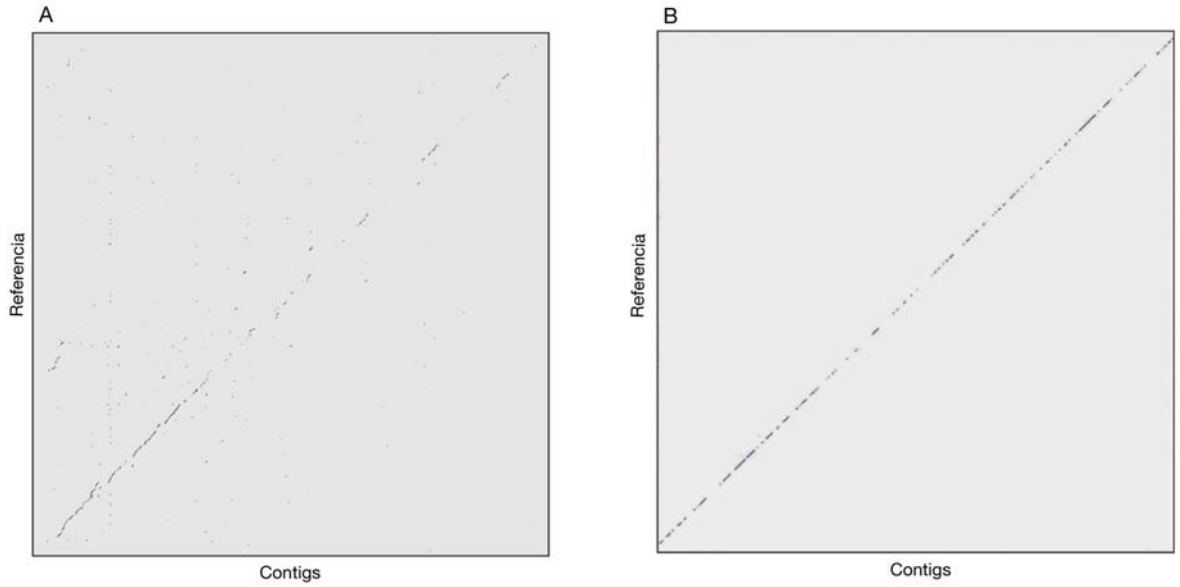

Figura IV.15*: Dot plot ejemplo de un alineamiento entre contigs genómicos reales de una cepa de la bacteria Lactococcus garvieae y una cepa de Lactococcus lactis (referencia). Antes del tratamiento con ICMapper (A) los fragmentos de alineamiento son pequeños y dispersos además que se nota la existencia de muchos alineamientos repetidos. Después del tratamiento ICMapper (B), los alineamientos resumidos aparecen más largos, de hecho son más representativos del contig ya que cubren zonas de la referencia más grandes, lo que permite un posicionamiento más fiable del contig. De otra parte se eliminaron los alineamientos repetitivos, los cuales probablemente fueron un producto de zonas repetitivas entre los dos genomas*

Para comprobar las posibilidades de ICMapper, se compararon sus resultados con los de 3 programas del mismo tipo, que son Oslay [154], Projector2 [155] y Mauve Aligner [156]. Los datos utilizados consistieron en borradores de genomas artificiales (contigs) creados a partir del genoma acabado de varias bacterias. En la creación de los contigs artificiales se incluyeron separaciones o solapamientos con longitudes aleatorias para tener una estructura de borrador de genoma parecida a los casos reales. Como referencia se utilizaron cepas de la misma especie o de especies diferentes del mismo género. La evaluación de la precisión del ordenamiento consistió en comparar el orden de los contigs establecido por los programas con su orden real ya conocido. En este estudio, para alinear los contigs artificiales contra el genoma de referencia se utilizó el la versión 2.2.20 de Blast con la siguiente configuración: -F F para desactivar el programa "DUST" de reconocimiento de secuencias de baja complejidad y evitar que se interrumpe el alineamiento Blast en estas zonas; -G 5, para afectar el valor 5 al coste por abrir huecos lo que permite reducir los huecos en los alineamiento, y -E 10 para afectar el valor 10 al coste por extender los huecos lo que permite reducir la longitud de los huecos abiertos. El resto de parámetros se dejaron por defecto.

Tal como se observa en la tabla IV.9, ICMapper y Mauve aligner generan mejores resultados que los dos demás programas, En caso de la especie *Mycobacterium tuberculosis* CDC1551, a la diferencia de los demás programas, ICMapper tuvo un porcentaje de ordenamiento de 100% en los dos casos donde la referencia era una cepa de la misma especie. En general ICMapper tuvo una precisión de ordenamiento superior o igual que Mauve aligner

y muy superior que Projector2 y Oslay, excepto el caso de la bacteria *Lactococcus lactis* subsp, cremoris MG1363, donde ICMapper tuvo una precisión de ordenamiento inferior a Mauve aligner (95.58% y 98.57% respectivamente).

***Tabla IV.9**: Porcentaje de pares de bases totales correctamente ordenadas utilizando ICMapper y tres otros programas. (B: Borrador de genoma, R: Referencia)*

			Suma de longitudes de los contigs correctamente ordenados por cada programa			
	Nº contigs artificiales	Suma de bases	ICMapper	Mauve aligner	Projector2	Oslay
B: *Mycobacterium tuberculosis* CDC1551 R: *Mycobacterium tuberculosis* H37Rv	144	4 311 730	4 311 730 100%	4 307 728 99,9%	4 292 255 99,54%	4 244 863 98,44%
B: *Mycobacterium tuberculosis* CDC1551 R: *Mycobacterium tuberculosis* KZN	144	4 311 730	4 311 730 100%	4 295 761 99,62%	4 290 321 99,50%	4 252 754 98,63%
B: *Mycobacterium tuberculosis* CDC1551 R: *Mycobacterium avium* 104	144	4 311 730	3 942 363 91,43%	3 919 573 90,90%	3 028 947 70,24%	3 549 569 82,32%
B: *Lactococcus lactis* subsp, cremoris MG1363 R: *Lactococcus lactis* subsp. lactis Il1403	88	2 497 652	2 387 292 95,58%	2 461 938 98,57%	2 087 813 83,59%	2 319 609 92,87%
B: *Streptococcus pneumoniae* 670-6B R: *Streptococcus pneumoniae* G54	77	2 197 707	2 122 247 96,56%	2 122 247 96,56%	1 993 147 90,69%	2 111 903 96,09%
B: *Bacillus amyloliquefaciens* DSM7 R: *Bacillus amyloliquefaciens* FZB42	140	3 920 736	3 888 585 99,17%	3 880 362 98,97%	3 234 464 82,49%	3 878 842 98,93%
B: *Bacillus thuringiensis* BMB171 R: *Bacillus thuringiensis* serovar konkukian str. 97-27	183	5 254 098	5 241 972 99,76%	5 074 150 96,57%	4 924 055 93,71%	5 139 721 97,82%
B: *Bacillus thuringiensis* BMB171 NC_014171 R: *Bacillus thuringiensis* str. Al Hakam	183	5 254 098	5 160 653 98,22%	5 176 617 98,52%	4 889 963 93,06%	5 007 363 95,3%
B: *Escherichia coli* 536 R: *Escherichia coli* 55989	154	4 795 594	4 667 799 97,33%	4 616 347 96,26%	4 023 816 83,9%	4 474 579 93,3%
Valor medio del porcentaje de los contigs correctamente ordenados			97,6%	97,3%	88,5%	94,9%

Por otra parte, se comparó el tiempo de ejecución de ICMapper con Mauve aligner (el programa que generó el resultado más parecido). La tabla IV.10 muestra el tiempo transcurrido durante todas las etapas del proceso (en caso de ICMapper, se incluye a tiempo transcurrido durante la etapa del alineamiento Blast). En todos los casos se observa una diferencia grande entre los tiempos de ejecución de los dos programas. ICMapper llegó a ser 20 veces más rápido que Mauve aligner. Esta diferencia en el tiempo de ejecución reside en el hecho de que Mauve aligner utiliza su propio algoritmo de alineamiento y ejecuta varios ciclos de ordenamiento después de los cuales escoge el mejor resultado basándose en criterios preestablecidos, mientras que ICMapper de una parte aprovecha de la rapidez del algoritmo Blast y de otra parte realiza la tarea de ordenamiento en un ciclo único y eficaz.

***Tabla IV.10**: Comparación entre los tiempos de todas las etapas de ejecución de los programas ICMapper y Mauve aligner*

	Tamaño del genoma	Tiempo global de ejecución (s)	
		ICMapper	Mauve aligner
Mycobacterium tuberculosis CDC1551	4 311 730	27	105
Lactococcus lactis subsp, cremoris MG1363	2 497 652	5	98
Streptococcus pneumoniae 670-6B	2 197 707	5	57
Bacillus amyloliquefaciens DSM7	3 920 736	7	61
Valor medio del tiempo de ejecución (s)		11	80

Contrariamente a Mauve aligner que esta destinado a genomas de tamaño pequeño, especialmente los genomas microbianos [156], ICMapper fue capaz de ordenar tanto borradores de genomas de microorganismos como borradores de genomas grandes de eucariotas. Ej. Genoma de *Solea senegalensis* (Apartado IV.3.2 a continuación)

IV.2. Ensamblajes de transcriptomas

IV.2.1. Estrategia tomada como modelo

Los ensamblajes y anotaciones del transcriptoma realizados en este trabajo se basaron sobre una flujo de trabajo que se desarrolló en nuestro grupo de investigación [87] para el transcriptoma del pino (*Pinus pinaster;* figura IV.16)

Preprocesamiento:

Las lecturas se preprocesan con SeqTrimNext, desarrollado también en nuestro laboratorio a partir de SeqTrim [152] (`http://www.scbi.uma.es/seqtrimnext/`; [46])

Ensamblaje:

A continuación se describe la estrategia del procesamiento y ensamblaje que se utilizó para el transcriptoma del pino (figura IV.17). Se pueden distinguir dos partes: ensamblaje, propiamente dicho, y reconciliación. El ensamblaje de lecturas largas se realiza con el método descrito en el apartado IV.1.2.3 basado en la combinación de los ensamblajes de MIRA [109] y EULER [110], el cual fue optimizado para extraer la máxima información de las lecturas y que ésta sea fiable. El ensamblaje de lecturas cortas se realiza con el programa ABySS [67] utilizando una serie de k-meros de 43 a 73. Los contigs de este ensamblaje se agrupan con CD-HIT [129] y después TGICL [157] a fin de eliminar las redundancias generadas principalmente por la utilización de múltiples k-meros. La reconciliación final consiste en un ensamblaje CAP3 [105] de los grupos de secuencias útiles procedentes de los ensamblajes anteriores que son los siguientes:

(i) Contigs de MIRA
(ii) *Debris* de MIRA que son codificantes (el resto se consideran artefactos de secuenciación)
(iii) Contigs de EULER sobre los que mapean lecturas
(iv) Contigs de EULER sobre los que no mapean lecturas, pero son codificantes (el resto se consideran artefactos de ensamblaje)
(v) Contigs agrupados de ABySS.

El ensamblaje da lugar a supercontigs y singulones que se juntarán para formar el transcriptoma final tal como se explicó más detalladamente en el apartado IV.1.2.3.

Verificación:

Este paso se realiza con Full-LengtherNext [87] que sirve para evaluar de un modo rápido si el ensamblaje tiene una buena proporción de genes completos, si está muy fragmentado o si posiblemente contiene un alto número de secuencias sin información

biológica. Esta verificación puede utilizarse tanto para evaluar la calidad del transcriptoma final como para evaluar la calidad de los ensamblajes de lecturas para poder optimizarlos con el ajuste de los parámetros de ensamblaje. Full-LengtherNext, después de las ultimas mejoras (Seoane et al., en preparación) tiene la posibilidad de crear un transcriptoma representativo que incluye los genes anotados únicos y los codificantes putativos. De este modo se reduce la fragmentación del transcriptoma que después resulta más manejable y se mejora la calidad de los resultados en los estudios posteriores (por ejemplo de RNAseq).

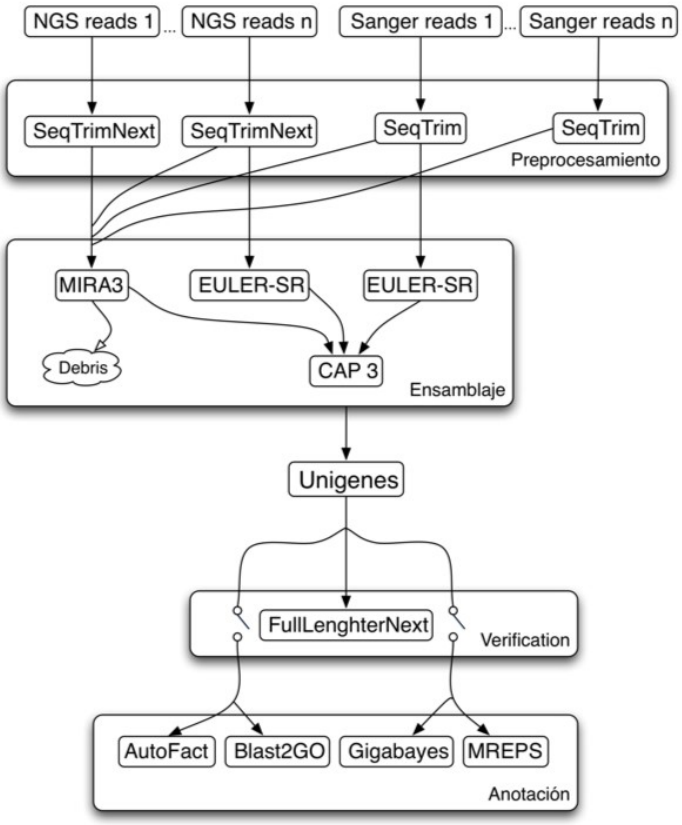

Figura IV.16: *(tomada de* [87]*): Flujo de trabajo propuesto para obtener un transcriptoma de pino anotado a partir de lecturas de NGS y de tipo Sanger. Las secuencias de tipo Sanger se preprocesan con SeqTrim y las de NGS con SeqTrimNext. Todas las lecturas limpias se ensamblan juntas con MIRA, y las secuencias limpias de NGS se ensamblan además con Euler-SR, pero sin mezclar los diferentes experimentos. Luego se re-ensamblan los contigs de Euler-SR y MIRA utilizando CAP3, posteriormente se verifican los unigenes del ensamblaje con Full-LengtherNext y si se considera que el ensamblaje es correcto, se anotan los unigenes obtenidos con AutoFact, Blast2GO, Gigabayes y MREPS*

Anotación:

La anotación que se propuso en el flujo (figura IV.16) se completa con los programas AutoFact, Blast2GO, Gigabayes y MREPS. De este modo, los transcritos quedan anotados con tres descripciones del producto del gen, que permiten a los investigadores confirmar si

tres programas que usan distintos criterios para anotación coinciden en la información encontrada. También se añaden términos GO, códigos EC, InterPro, rutas KEGG, SNP y SSR que permitirán realizar estudios computerizados de los resultados con programas de enriquecimiento biológico o detección de posibles marcadores moleculares para mejora genética.

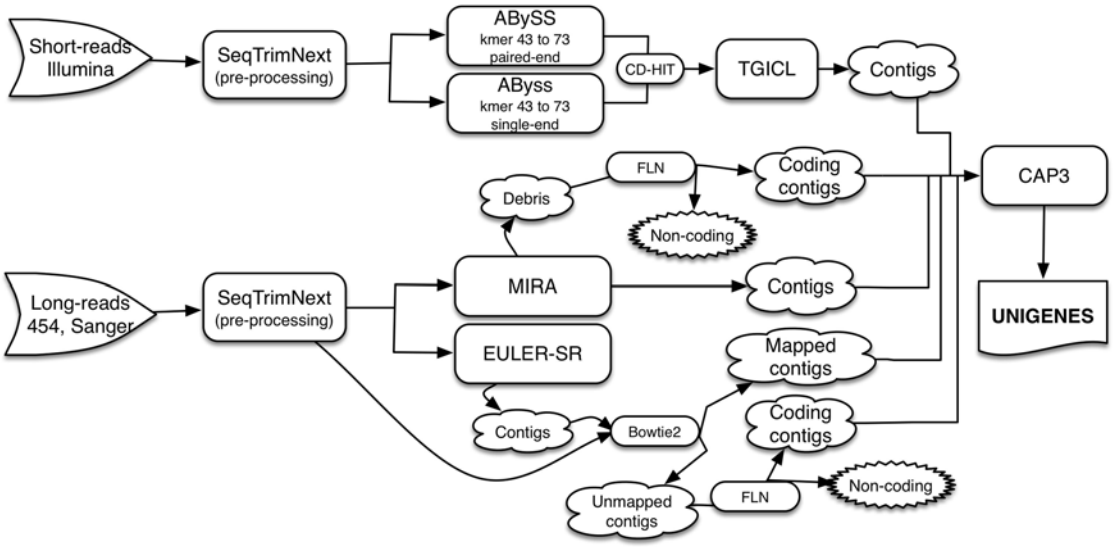

Figura IV.17: *(tomada de [158]) Esquema del flujo de trabajo del procesamiento, ensamblaje y reconciliación de las lecturas 454 y Illumina de Pinus pinaster*

Generación de una base de datos:

El primer prototipo de base de datos transcriptómica desarrollada por nuestro laboratorio fue EuroPineDB [141], que poco después fue mejorada y ampliada en SustainPine [158] (figura IV.18). Están basadas en Ruby On Rails tal y como se describe en material y métodos (apartado I.3.4.1). La principal mejora entre las dos está en que la web, la base de datos propiamente dicha, y los cálculos se alojan en una estructura de 3 máquinas virtuales [46] para agilizar su uso. También se mejoró el sistema de paginación interna y la forma de navegar por el contenido a base de tabuladores interactivos.

Figura IV.18: *Base de datos de SustainPineDB en su estado inicial. A: Arquitectura de la base de datos. B: Página de inicio. C: página de anotaciones*

IV.2.2. Transcriptomas de *Solea senegalensis* y *Solea solea*

El lenguado senegalés (*Solea senegalensis*) y el lenguado común (*Solea solea*) son dos especies de peces planos importantes desde el punto de vista económico y evolutivo, tanto en la pesca como en la acuicultura. A pesar de la disponibilidad de algunos recursos genómicos que se describieron recientemente, se necesitan aún más experimentos de secuenciación para establecer un transcriptoma completo y representativo. Por otra parte, un análisis comparativo del transcriptoma de ambas especies puede ayudar a entender la evolución de pez plano.

IV.2.2.1. Preprocesamiento de las librerías

Un total 37 y 43 librerías de Illumina de *Solea senegalensis* y *Solea solea* respectivamente (1 800 millones de lecturas) y 14 librerías adicionales de Roche/454 para *Solea senegalensis* (5,6 millones de lecturas) fueron preparadas en el laboratorio del grupo de investigación de IFAPA El Toruño (Cádiz). En la tabla IV.11 se presenta un resumen del preprocesamiento con SeqTrimNext de las lecturas Roche/454 e Illumina de todas las librerías que se pusieron a nuestra disposición. Se observa que la mayoría de las lecturas pareadas fueron útiles (83,3% y 79,5% en *S. senegalensis* y *S. solea* respectivamente). La fuente de contaminaciones más importantes fue de secuencias ribosómicas y mitocondriales. Otros contaminantes menos representados en los datos de Illumina fueron secuencias de los organismos utilizados para la nutrición de las larvas (artemias (8-21%) y rotiferos (2-4%), principalmente) y otros microorganismos diferentes, principalmente de hongos.

Tabla IV.11: *Resumen del pre-procesamiento de las lecturas originales de los transcriptomas de S. senegalensis y S. solea*

			NGS platform		
	Reference to Figure 1		Illumina		454
Species			*S. senegalensis*	*S. solea*	*S. senegalensis*
Total Input Reads	#1		1,800,249,230	2,101,324,072	5,663,225
Mean length			**76**	**100**	**757**
Rejected (total)	#2	N	237,941,945	345,251,849	1,562,661
		%	13.5	17.1	26.8
By contamination		N	144,247,943	226,627,909	156.921
		%	8.2	11.2	3.0
Useful reads	#3		1,561,416,814	1,746,258,741	3,774,412
			86.7	83.1	67.6
Paired reads		N	1,503,882,050	1,676,160,406	-
		%	83.3	79.5	-
Single reads		N	57,534,764	70,098,335	3,774,412
		%	3.2	3.3	67.6
Mean length			**66**	**89**	**184**

Suponiendo que el número de los genes putativos de *Solea* puede variar entre 21 516 y 26 206 genes codificantes a proteínas como se describió en el Lenguado a lengua (*Cynoglossus semilaevis*) y el pez cebra (*Danio rerio*) respectivamente [159, 160], con una longitud media del transcrito de 2 841 nt, la cobertura estimada de las lecturas útiles estuvo entre 1 384x y 1 686x para *S. senegalensis* y entre 2 088 y 2 543 para *S. solea*.

IV.2.2.2. Ensamblaje

Ensamblaje de *Solea senegalensis*

Se hicieron varias versiones del transcriptoma de *S. senegalensis* utilizando diferentes datos de Roche/454 a medida que iban estando disponibles, con el protocolo ilustrado en la figura IV.19, que es el mismo que se utilizó para ensamblar las lecturas largas del pino. La mejor versión fue la 3.0, obtenida con todas las lecturas largas de la tabla IV.11. Esta versión permitió iniciar la construcción de una micromatriz para solea, así como estudiar la expresión de genes de interés en el IFAPA El Toruño por el grupo de Manuel Manchado (datos no mostrados).

Para mejorar el ensamblaje del transcriptoma (y obtener la versión 4), se utilizó el flujo de trabajo ilustrado en la figura IV.20 donde se añadieron los datos de Illumina que se habían obtenido para otros experimentos de desarrollo embrionario y respuesta a hormonas. En este flujo de trabajo, la estrategia de lecturas largas fue ligeramente modificada (esencialmente, todos los *debris* de MIRA fueron descartados, dado que íbamos a contar con suficientes secuencias cortas que no se considerarían artefactuales).

En el ensamblaje de las lecturas cortas no se hizo con ABySS sino con Oases [115]. La elección de este programa se basó sobre un estudio reciente [161] en el cual se compararon varios programas de ensamblaje de lecturas cortas transcriptomicas y colocó a Oases MK (multi k-meros) como el que cubría el mayor número de genes y el mayor número de genes completos. En el ensamblaje preliminar de lecturas cortas con Oases MK utilizando 5 k-meros (23, 29, 35, 41 y 47) se obtuvo un número alto de transcritos (736 100) lo que fue esperado teniendo en cuenta la gran variabilidad motivada por la combinación de lecturas procedentes de varias muestras y muchos millones de individuos diferentes. Es de esperar que este elevado número contenga no solo parálogos de los genes, sino también alelos, quimeras, forma alternativas de ayuste y transcritos sin terminar de madurar. Como nuestro objetivo fue tener un transcriptoma con un número reducido de transcritos, pocas duplicaciones y un buen porcentaje de mensajeros completos, realizamos una comparación este ensamblaje donde se utilizaron 5 k-meros con otro donde se utilizaron solo el k-mero más grande y más pequeño (23 y 47). En la tabla IV.12 se muestran algunos índices claves de calidad del ensamblaje generado en cada caso, que permiten determinar que al utilizar solo los dos k-meros 23 y 47, el número de transcritos totales pasó de 736 100 a 514 888. Esta disminución del número de transcritos se acompañó con una disminución del número de ensamblajes erróneos que pasó de 403 a 150 y del número de quimeras que pasó de 47 224 a 20 406. De otra parte, fue interesante notar que aunque hubo una disminución de número de transcritos con ortólogo (de 195 918 a 121 608), no se afectaron tanto los números de ortólogos diferentes y completos

diferentes, considerados como los índices de calidad más pertinentes (que infravaloran los transcritos sin madurar completamente y artefactuales), ya que pasaron de 48 328 a 42 287 y de 15 174 a 13 490 respectivamente. Como los ensamblajes con los k-meros intermedios (29, 35 y 41) no parecían aportar un número significativo de genes adicionales ni mejorar de forma significativa la calidad de los transcritos anotados, decidimos quedarnos con el ensamblaje con los k-meros 23 y 47 priorizando tener menos cantidad de transcritos, menos ensamblajes erróneos y menos quimeras.

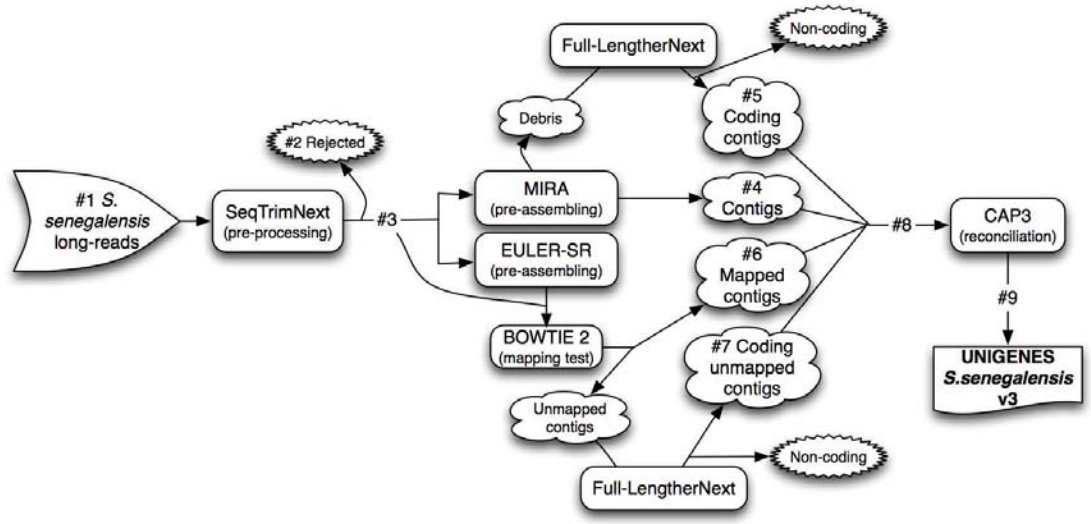

Figura IV.19: *(tomada de [143]) Estrategia de pre-procesamiento, ensamblaje y la reconciliación de las versiones 1 a 3 del transcriptoma de S. senegalensis*

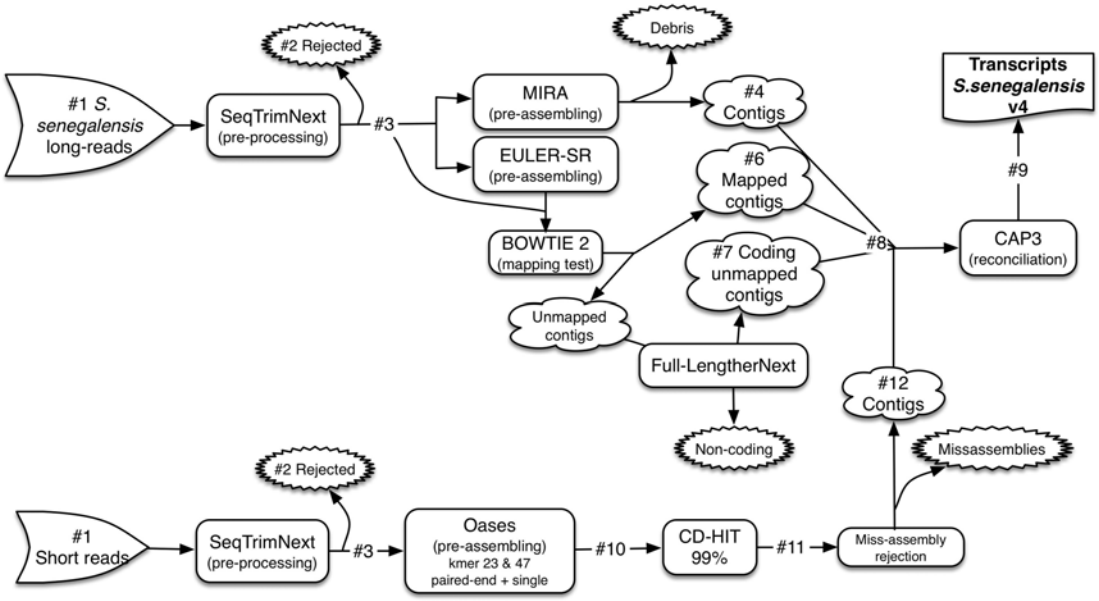

Figura IV.20: *(tomada de [143]) Estrategia de pre-procesamiento, ensamblaje y la reconciliación de la versión 4 del transcriptoma de S. senegalensis*

Tabla IV.12: *Ensamblaje de las lecturas cortas de S. senegalensis con diferentes conjuntos de k-meros*

	K-meros	
	23, 29, 35, 41, 47	**23 y 47**
Scaffolds	736 100	514 888
Media de longitudes	639	592
Ensamblajes erróneos	403	150
Quimeras	47 224	20 406
Ortólogos	195 918	121 608
Ortólogos diferentes	48 328	42 287
Completos	34 284	23 354
Completos diferentes	15 174	13 490
Codificantes	16 518	10 491

Después de agrupar los transcritos de Oases con CD-HIT, se añadió un paso de validación donde se comparan los transcritos procedentes de lecturas cortas con aquellos procedentes de las lecturas largas que se consideran más fiables debido a la longitud más grande de las lecturas y a que el algoritmo utilizado para ensamblarlos es de tipo OLC. Esta comparación consiste en un alineamiento Blast que luego se recorre con un script Ruby para detectar los transcritos con una estructura sentido/antisentido y poder eliminarlos (figura IV.21).

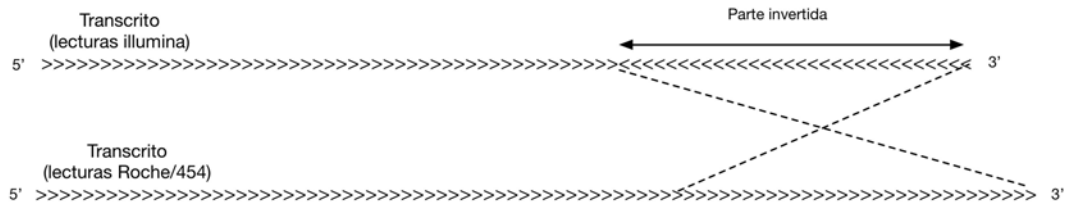

Figura IV.21: *Utilización de los transcritos procedentes de las lecturas Roche/454 para detectar las estructuras sentido/antisentido en los transcritos de Illumina*

Ensamblaje de *Solea solea*

Para el ensamblaje de *S. solea* (figura IV.22) se utilizó la misma estrategia de ensamblaje de lecturas cortas de la versión 4 de *S. senegalensis* (figura IV.20) con una diferencia respecto a la parte inferior de la figura en los valores de k-meros, ya que en vez de 23 y 47, se utilizaron 25 y 69 por la razón de que lecturas brutas eran más largas (tabla IV.11).

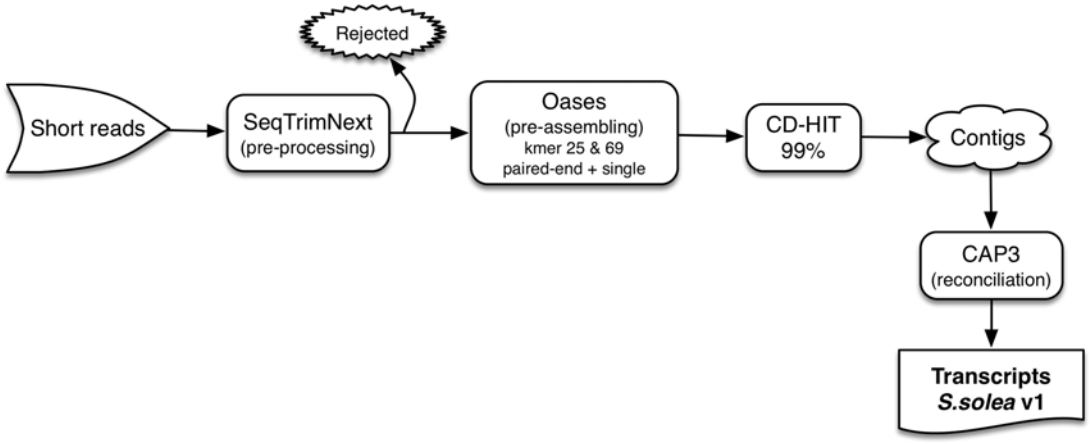

Figura IV.22: *Esquema de ensamblaje del transcriptoma de S. solea*

Balance de los transcriptomas ensamblados

La cronología de las versiones de los transcriptomas de *S. senegalensis* y *S. solea* según los tipos de lecturas y/o tejidos utilizados se presenta en la tabla IV.13. Los transcriptomas de *S. senegalensis* v4 y *S. solea* v1 fueron similares con respecto a (i) frecuencia de distribución de las longitudes de transcritos (figura IV.23), (ii) número total de transcritos y (iii) número de transcritos superiores a 500 pb. Sin embargo, la longitud media y N50 fueron claramente superiores en *S. solea*, lo que se puede explicar por las lecturas de entrada más largas (89 contra 66 en *S. solea* y *S. senegalensis* respectivamente; tabla IV.11) y la baja contribución de las lecturas Roche/454 en *S. senegalensis*. Esta baja contribución puede ser explicada por el hecho que las librerías Roche/454 fueron normalizadas para reducir los transcritos altamente abundantes, lo que podría haber dado lugar a ensamblajes más fragmentados limitando la contribución las lecturas Roche/454 en el transcriptoma final [162]

Tabla IV.13: *Visón general de las diferentes versiones de S. senegalensis y S. solea según el tipo de lecturas y/o tejidos utilizados. MMC: sistema inmune, CIT: Osmoregulación, HPP: Hipófisis, HPT, Hipotálamo, GND: Gónadas*

Versión	Tipo de lecturas	Tejidos	Nº transcritos
Solea senegalensis v1.0	Roche/454	MMC, CIT	136 448
Solea senegalensis v2.0	Roche/454	MMC, CIT, HPP, HPT, GND	250 132
Solea senegalensis v3.0	Roche/454	MMC, CIT, HPP, HPT, GND	252 416
Solea senegalensis v4.0	Roche/454	MMC, CIT, HPP, HPT, GND	697 125
	Illumina	Multi-tejido	
Solea senegalensis v4.1	Roche/454	MMC, CIT, HPP, HPT, GND	694 279 (total) 59 514 (referencia)
	Illumina	Multi-tejido	
Solea solea v1.0	Illumina	Multi-tejido	523 637
Solea solea v1.1	Illumina	Multi-tejido	531 463 (total) 54 005 (referencia)

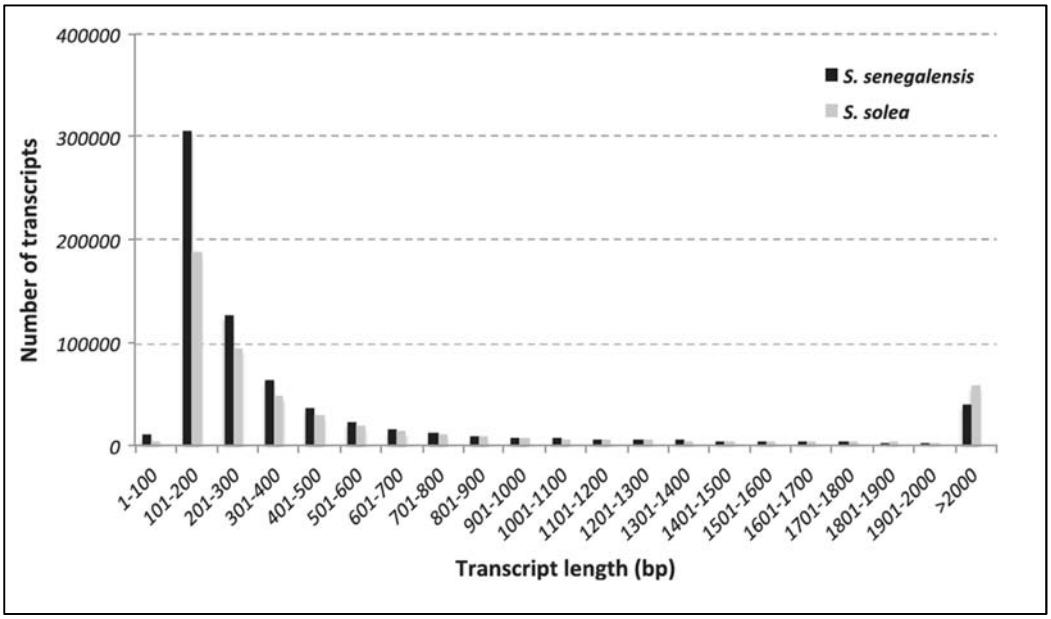

Figura IV.23: *Representación de los transcritos abundantes con respecto a sus longitudes en los transcriptomas de S. senegalensis (barras negras) y S. solea (barras gris)*

IV.2.2.3. Anotación de los transcriptomas.

Los transcriptomas de *S. senegalensis* y *S. solea* fueron anotados según el mismo flujo de trabajo general de la anotación del pino, si bien se introdujeron algunos cambios para mejorar la eficiencia de la anotación y para adaptarse a los nuevos tipos de lecturas utilizadas. Por ejemplo, debido a la gran cantidad de transcritos obtenidos en *S. senegalensis* y *S. solea*, no fue posible utilizar Blast2GO por ser muy lento, por lo que puede llevar meses anotar estos transcriptomas. Blast2GO fue reemplazado por otro programa desarrollado en nuestro laboratorio: Sma3s [84] (véase apartado III.3.4.5 de Materiales y métodos), que tiene una fiabilidad demostrada de anotación parecida y que además es totalmente paralelizable, lo que permitió anotar los dos transcriptomas en relativamente poco tiempo.

Por otro lado, se añadieron dos nuevos elementos de la anotación de los transcritos. El primero fue el ID de RefSeq y de Ensembl de una especie modelo al ser un elemento muy útil en los estudios funcionales, comparativos y de expresión. La afectación de estos ID a los transcritos se realiza con un Blastx del transcriptoma contra el conjunto de proteínas RefSeq y ENSEMBL descargados desde las webs NCBI y ENSEMBL, tomando como filtros mínimos un valor de $E = 10^{-10}$ y un porcentaje de identidad del 30%. Para las dos especies de lenguado se utilizó el pez cebra (*Danio rerio*).

Otro nuevo elemento de anotación añadido son los miRNA. Así, se definieron los transcritos más susceptibles de ser un precursor de miRNA realizando un Blast del transcriptoma contra la base de datos GAmiRdb. Esta lista de transcritos puede ser objeto de estudios más profundos para fiabilizar los miRNAs contenidos en ellos.

Respecto a la detección de los SNP, no fue posible utilizar Gigabayes ya que no era compatible con los ensamblajes de las lecturas cortas especialmente por el tipo de algoritmo utilizado para ensamblarlas. Los programas de detección de SNP adaptados a lecturas cortas utilizan la información del mapeo de esas lecturas sobre el transcriptoma para detectar la variación vertical de los nucleótidos en las lecturas alineadas respecto al nucleótido de referencia (del transcrito sobre el cual mapean). El primer programa que probamos en este propósito fue SOAPsnps (`http://soap.genomics.org.cn/soapsnp.html`), el cual fue descartado porque generaba poca información útil de los SNP y no presentaba sus valores de calidad. Otro programa probado fue Atlas-SNP [163] que mostró una información más completa que incluía sus valores de calidad, y las presentó en un fichero tabulado con el formato estándar VCF (*variant call format* [128]). Pero preferimos la herramienta mpileup [164] del paquete Samtools que genera un fichero VCF como Atlas-SNP, y además incluye a los porcentajes de las frecuencias del nucleótido de la referencia (del transcrito) y del nucleótido alternativo.

El nuevo flujo de anotación utilizado para los dos transcriptomas de lenguados donde se incluyen los nuevos elementos y programas de anotación se muestra en la figura IV.24. Este flujo puede ser utilizado para cualquier ensamblaje de transcriptomica con lecturas Illumina. Las diferentes anotaciones generadas en este flujo se subirán a la base de datos web descrita en el siguiente párrafo.

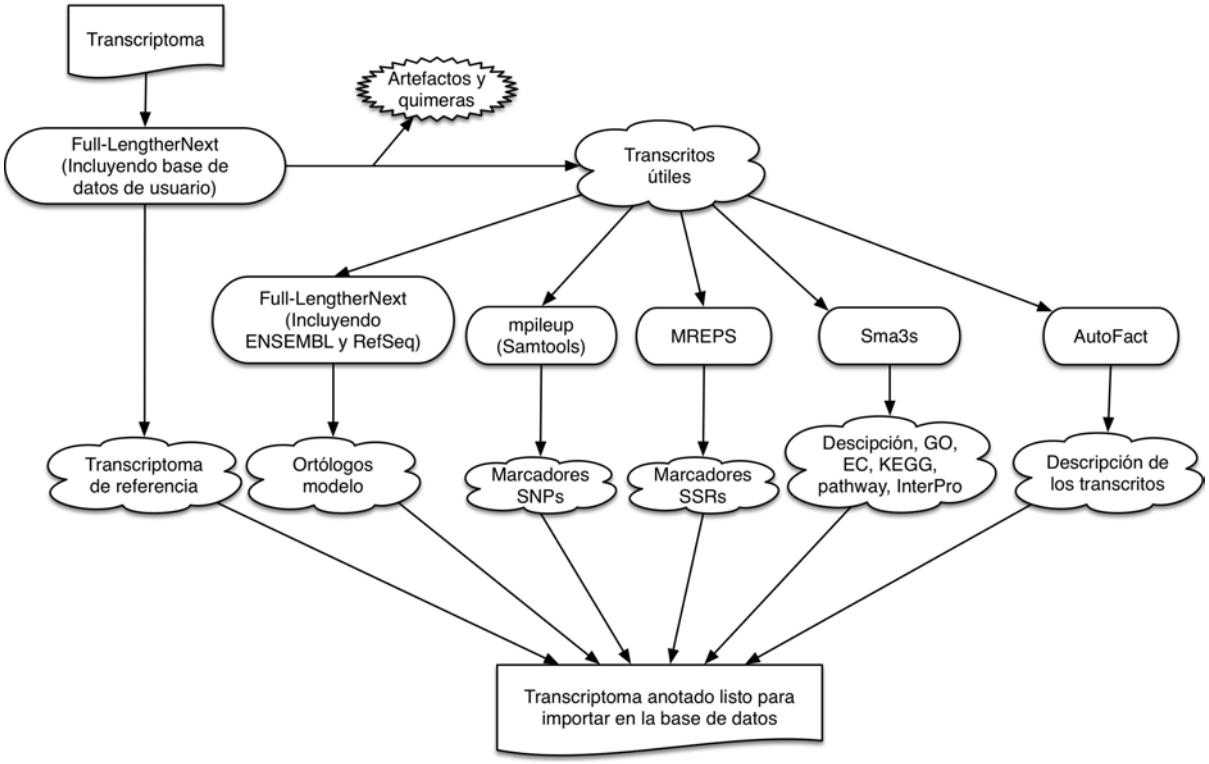

Figura IV.24: *Flujo de trabajo de la anotación de los transcriptomas de lenguado*

IV.2.2.4. SoleaDB, una base de datos para explorar los transcriptomas de Solea

SoleaDB fue construida sobre la base de SustainPineDB para almacenar los distintos ensamblajes de los transcriptomas de *S. senegalensis* y *S. solea*, y los diferentes tipos de anotaciones.

En la figura IV.25 se puede observar el esquema de las tablas de la base de datos en SoleaDB y en la figura IV.26 se muestran unas capturas de pantalla de la interfaz web. Las principales mejoras de SustainPineDB aplicadas en SoleaDB fueron las siguientes: (i) Mejora en el modo de acceso a los datos de anotaciones. En efecto, ahora se puede acceder a las anotaciones específicas a cada versión del transcriptoma en una estructura con pestañas horizontales que permiten visualizar de forma independiente la información relativa a cada versión del transcriptoma (figura IV.26-B). (ii) Se añadieron estadísticas completas sobre cada ensamblaje y sus anotaciones, la cual se puede consultar en la página "Assembly info". (iii) En la lista de transcritos, se añadió la posibilidad de buscar por nombre de transcritos o por anotaciones y de otra parte se implementó una función que permite resaltar las palabras claves comunes entre las descripciones de los 3 programas de anotación, lo que permite al usuario de localizar fácilmente las descripciones comunes. (iv) En la página donde se presenta la información sobre el transcrito, se añadió un campo de curación donde se pueden introducir notas o información sobre el transcrito en cuestión. (v) En la página de la información sobre el ensamblaje, se implementó una herramienta que permite extraer toda la

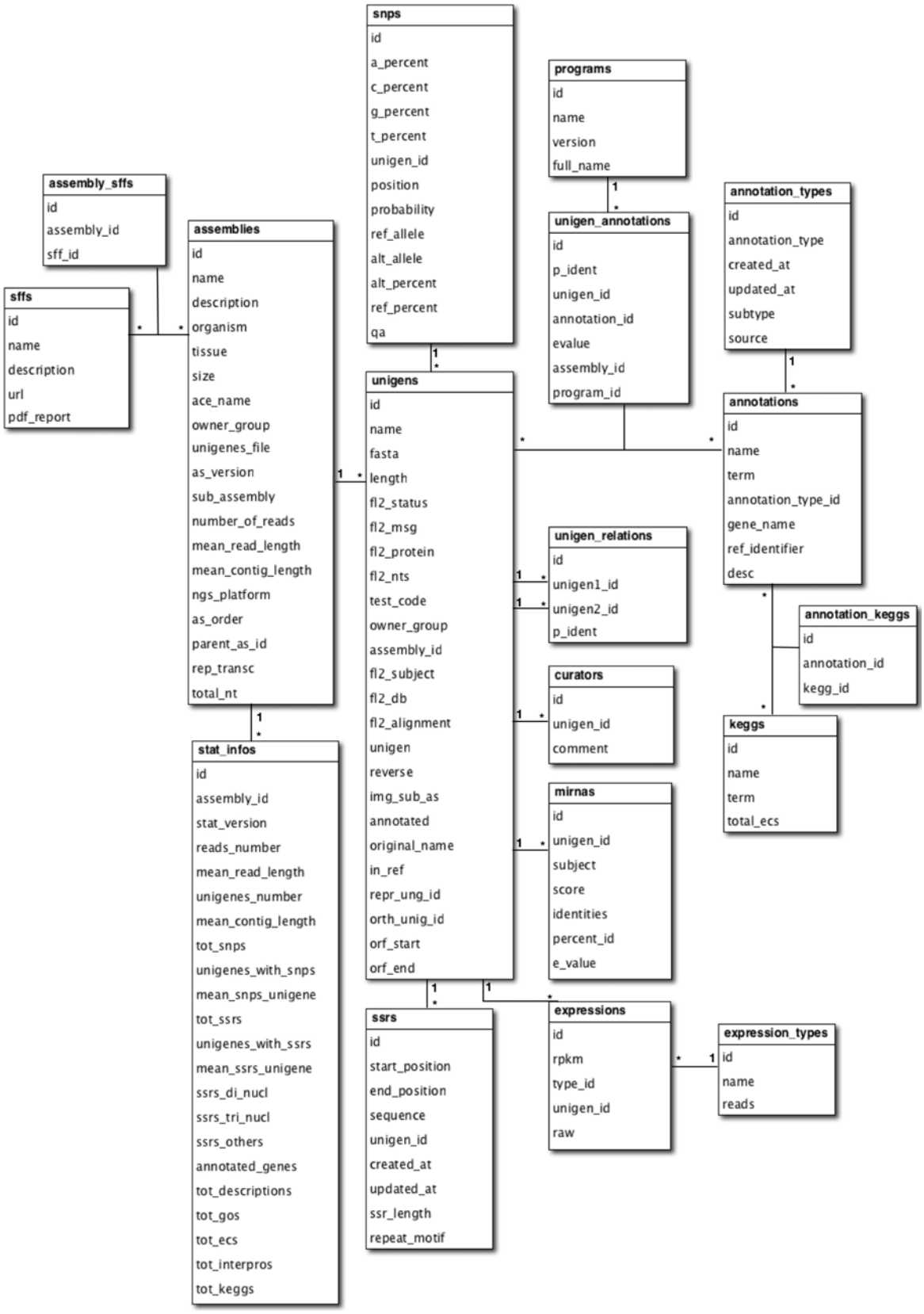

Figura IV.25: *Esquema de las tablas de base de datos SoleaDB que guardan las misma estructura utilizada en SustaPineDB con algunos elementos adicionales*

información útil sobre una lista de transcritos y exportarla a un fichero en formato tabulado, el cual se puede abrir con programas como Excel (figura IV.26-C). (vi) En la página de los KEGG se incorporó la posibilidad de visualizar las mapas de las rutas metabólicas en las cuales se resaltan en color verde los códigos ECs propios de la especie estudiada. En la figura IV.27 se puede observar un ejemplo de los mapas KEGG a los que se puede acceder desde la web.

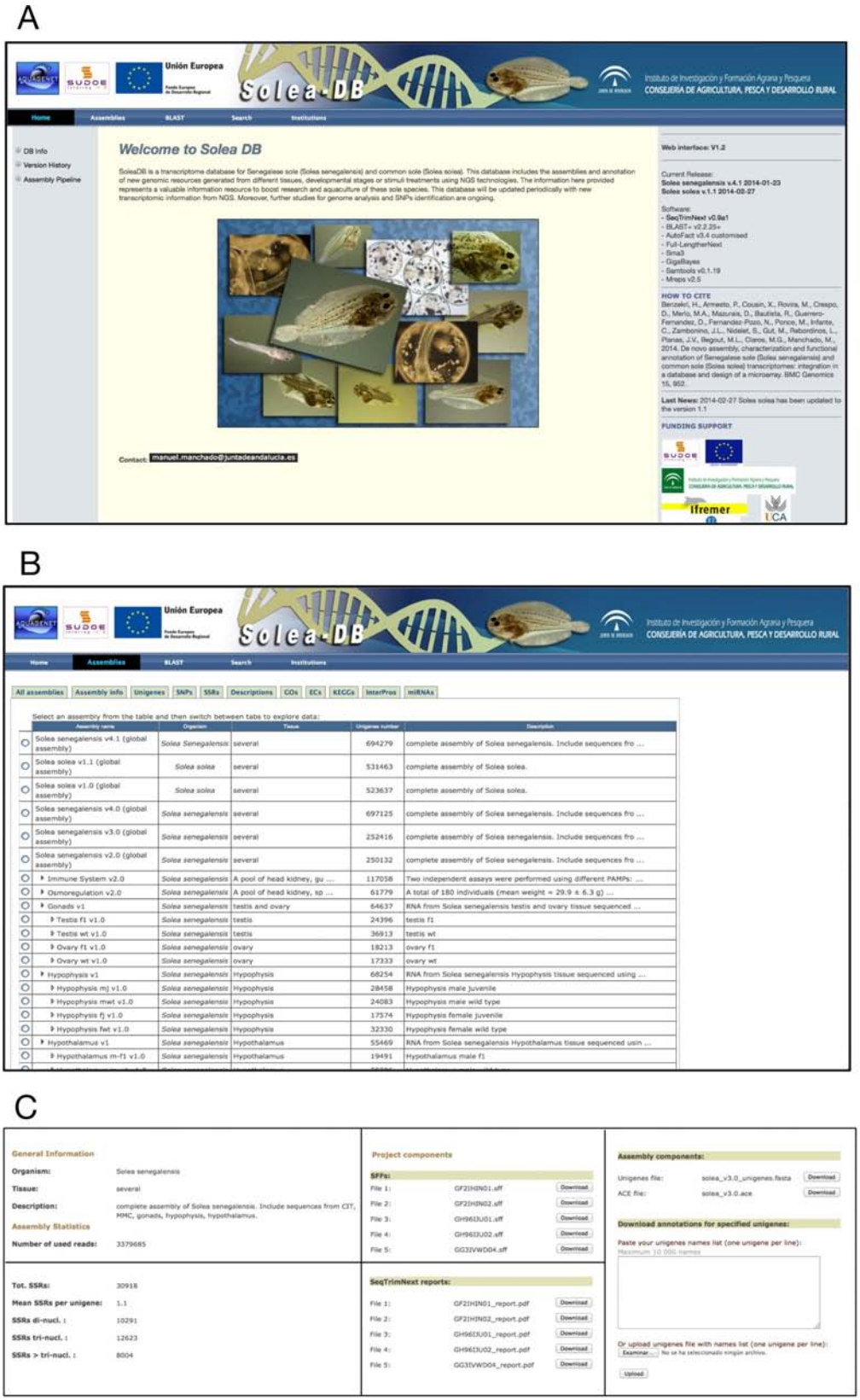

Figura IV.26: *Capturas de pantalla de la interfaz de la base de datos SoleaDB. A: Portada de la web. B: Ilustración de la pestaña "Assemblies" en la que se muestra toda la información sobre todas las versiones de transcriptomas y subversiones. C: captura de la parte de pantalla que corresponde a la pestaña "Assembly info" donde hay información general sobre el transcriptoma, así como ficheros descargables y herramientas*

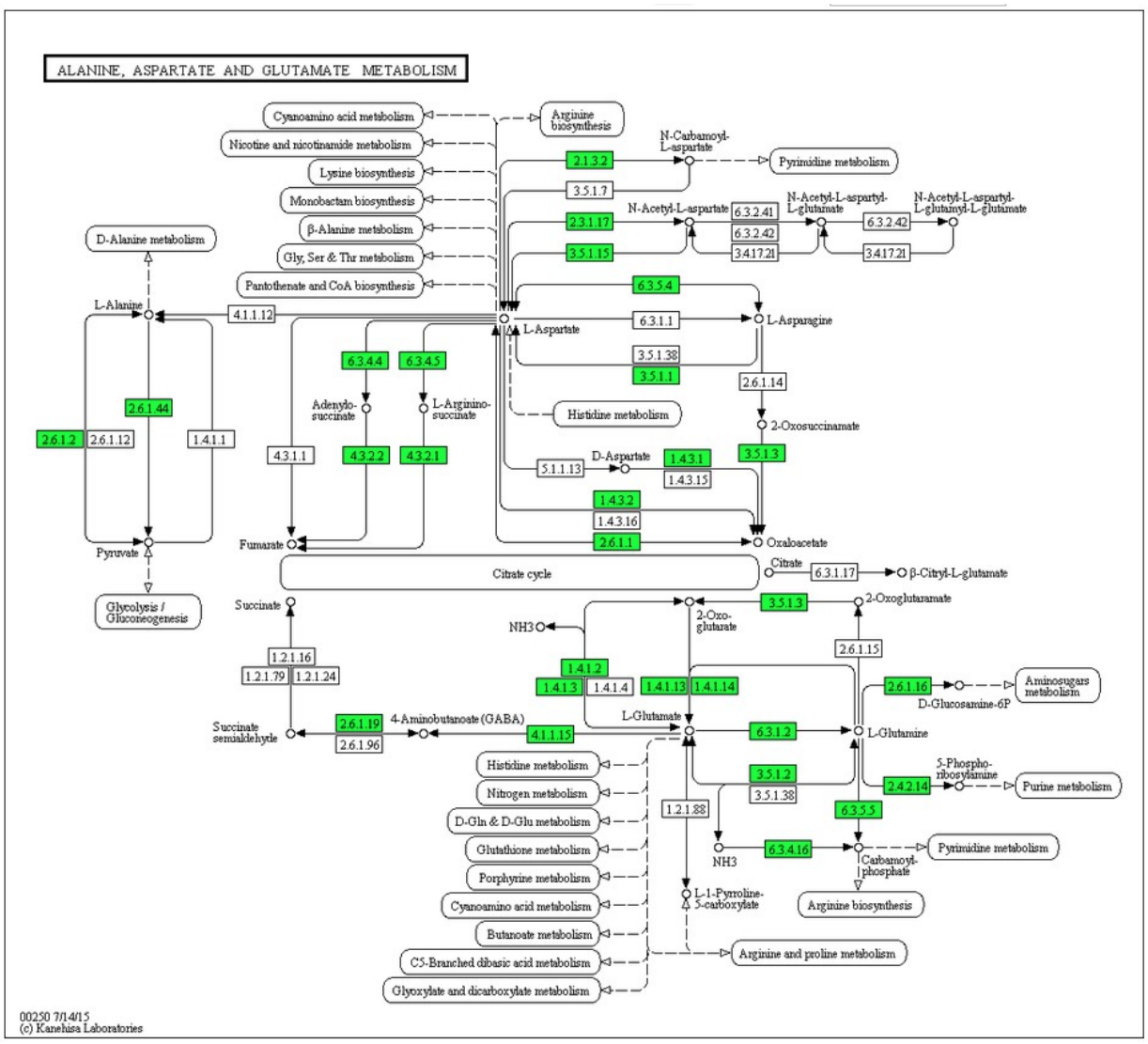

Figura IV.27: *Mapa de KEGG de la ruta del metabolismo de alanina, aspartato y glutamato, en el que las enzimas anotadas en S. senegalensis aparecen resaltadas en verde.*

IV.2.2.5. Calidad del transcriptoma de los dos lenguados

Un análisis detallado del resultado de anotación Full-LengtherNext (tabla IV.14) revela primero que, a pesar del número muy elevado de los transcritos, el número de artefactos fue muy bajo. Segundo, que el transcriptoma de *S. senegalensis* v4 mejoró con respecto a la v3 en términos longitud de los transcritos y número de los ORF completos, aunque ambas versiones presentaron un número similar de ID diferentes de ortólogos, lo que muestra la importancia de utilizar muchos tejidos para la preparación de las librerías Roche/454 a la hora de obtener una mayor representación de los genes. Tercero, el número de los ORFs completos diferentes fue alto y similar entre los transcriptomas de las dos especies (tabla IV.14 "Diferent complete ORFs"), poniendo de manifiesto que los transcriptomas de lenguado son igual de fiables. Finalmente, *S. senegalensis* v4 tuvo menor número de ID de ortólogos y menor número de ORF completos diferentes que *S. solea* v1 lo que puede ser debido a una mayor

fragmentación en *S. senegalensis* ya que, según parece, los datos de Roche/454 solo contribuyeron en incrementar la ortología en el ensamblaje final.

Tabla IV.14: *(tomada de [143]) : Características de los transcriptomas ensamblados de S. senegalensis y S. solea*

	S. senegalensis				*S. solea*	
	v3		*v4*		*v1*	
	Transcripts	**%**	**Transcripts**	**%**	**Transcripts**	**%**
Transcripts	252,416	-	697,125	-	523,637	-
Artifacts[2]	nc	-	7,095	1.02	10,086	1.92
Valid transcripts	252,416	100.00	701,767	100.00	531,463	100.00
>500pb	37,593	14.90	156,083	22.24	165,860	31.22
>200pb	168,914	66.92	385,411	54.92	338,967	63.89
Longest transcript	6,050	-	40,163	-	30,526	-
Transcripts with ortholog[1]	81,348	32.23	147,536	21.02	121,696	22.90
Different orthologous IDs	41,792	51.37	45,063	30.54	38,402	31.56
Complete ORFs	6,742	8.31	39,727	26.93	52,051	42.77
Different, complete ORFs	4,376	5.38	18,738	12.70	22,683	18.64
C-terminus	14,757	18.14	27,080	18.35	19,579	16.09
N-terminus	11,298	13.88	27,638	18.73	25,131	20.65
Internal	47,529	58.43	53,091	35.99	24,935	20.49
Putative ncRNAs	539	0.21	1,252	0.18	1,075	0.20
Transcripts without ortholog[1]	171,067	67.56	545,491	77.73	408,692	76.90
Putative new transcripts	22,612	13.21	39,812	7.30	34,194	8.37
Non-redundant new transcripts	nc	–	14,451	2.65	15,603	3.55
Unknown	147,916	86.48	505,679	92.70	374,498	91.63
Reference transcriptome	nc	–	*v*4.1: **59,514**	8.48	*v*1.1: **54,005**	10.16

The values were calculated using FullLengtherNext software. The minimum number of transcripts that can be considered as a reference transcriptome is shown in bold.
[1]Percentages for subclassifications of this category were calculated using this line as 100% reference.
[2]Artifacts refer mainly to misassemblies and chimeric contigs.
nc: Non-calculated.

El alto número de lecturas utilizadas en el ensamblaje de los dos transcriptomas de lenguado puede haber favorecido la acumulación de errores [162, 165]. La evaluación de la fiabilidad del transcrito se basó inicialmente en el mapeo de las lecturas útiles de dos librerías aleatorias de cada transcriptoma utilizando Bowtie2 [119]. Como el 96,7%-98% de las lecturas mapearon sobre los transcritos, los errores de ensamblaje pueden considerarse insignificantes. Curiosamente, el transcrito más largo de los transcriptomas de *S. senegalensis* v4 y *S. solea* v1 no es un artefacto, ya que en ambas especies corresponde a una proteína parecida a la titina cuyo ARNm es muy largo (94 446 bp) y que anteriormente ya se ha ensamblado en la tilapia (Número de acceso XM_005460929). El hecho de que este transcrito es 6 veces más largo en *S. senegalensis* v4 que en *S. senegalensis* v3 supone la contribución importante de las lecturas cortas en el ensamblaje final.

Los transcritos sin ortólogo como fuente de nuevos transcritos de lenguado.

El alto número de transcritos sin ortólogo (tabla IV.14) mereció un análisis más profundo. En función del índice de testcode [135], más del 91% de los transcritos son desconocidos ("Unknown"), lo que puede explicar en buena parte la falta de ortología. Para

verificar el origen de estos transcritos desconocidos, se mapearon lecturas genómicas (desde varias librerías genómicas de *S. senegalensis* disponibles en nuestro laboratorio) sobre los transcritos desconocidos de *S. senegalensis* v4, lo que dio lugar a 462 568 (91,25%) transcritos desconocidos mapeados (440 385 con más de 10 lecturas). Este alto porcentaje de mapeo indica que estas secuencias no fueron artefactos de ensamblaje y que pueden corresponder a fragmentos genómicos o transcritos inmaduros que copurifican con los transcritos maduros. Por otra parte, entre 7,30% y 8,37% de los transcritos sin ortólogo en *S. senegalensis* v4 y *S. solea* v1, respectivamente, algunos mostraron un índice Testcode >0,94 (tabla IV.14, "Putative new transcripts"), por lo que tienen alta probabilidad de ser transcritos codificantes. La fracción no redundante de estos transcritos (14 451 y 15 503 transcritos en *S. senegalensis* v4 y *S. solea* v1, respectivamente; tabla IV.14), sobre la base de la ausencia de ortólogos en la base de datos UniProtKB, salvo que sean contaminantes de especies no secuenciadas aún, pueden considerarse proteínas (o fragmentos de proteínas) "nuevas" de lenguado.

Transcriptoma de referencia en cada especie de lenguado

El número alto de transcritos ensamblados indicó una sobreestimación de los transcriptomas de lenguado cuando se comparan con otros teleósteos [159, 160]. Probablemente, la sobrevaloración venga de los transcritos que pueden representar alelos, parálogos, transcritos fragmentados, isoformas o secuencias alternativas de ayuste, ARNm inmaduros o incluso una combinación de ellos. Por lo tanto, para los estudios de estos transcriptomas conviene utilizar el transcriptoma de referencia generado por Full-LengtherNext que incluye un número de transcritos mucho más reducido y cercano al teórico. El transcriptoma de referencia para *S. senegalensis* (*S. senegalensis* v4.1) consistió en 59 514 transcritos y para *S. solea* (*S. solea* v1.1) consistió en 54 005 transcritos (tabla IV.14, última fila)

Para verificar la representatividad de los transcritos que constituyen los transcriptomas de referencia, se mapearon las lecturas útiles (tabla IV.11) sobre los dos transcriptomas de referencia. Como resultado, el 82,3-87,5% de las lecturas mapearon sobre los transcritos, mientras que el 76,5-93,3% de los transcritos recibieron el mapeo de más de una lectura, lo que muestra que estos transcritos representan adecuadamente el transcriptoma. Otra verificación del transcriptoma de referencia (v4.1 y v1.1) se basó en un análisis de ortología con el pez cebra (43 132 entradas disponibles en RefSeq y 42 555 en ENSEMBL) utilizando Top-blast. En *S. senegalensis* v4.1, 39 851 transcritos de referencia fueron ortólogos a 21 542 entradas RefSeq y 20 753 entradas Ensembl de pez cebra (figura IV.28, izquierda). En *S. solea* v1.1, 34 949 transcritos de referencia fueron ortólogos a 20 594 entradas RefSeq y 19 632 entradas Ensembl de pez cebra (figura IV.28, derecha). Estos números muestran que un cierto número de alelos, ARNm inmaduros y genes específicos de linaje (incluso algunos transcritos quiméricos no detectados) pudieron haber sido incluidos en el transcriptoma de referencia. Por otra parte, como el número de ID de RefSeq y Ensembl corresponde casi a la mitad de los transcritos de lenguado, es probable que se hayan incluido ambos alelos de cada gen en el transcriptoma de referencia. Esta hipótesis también está soportada por el hecho de que las muestras analizadas corresponden a animales silvestres y a larvas, que son

mayoritariamente heterocigotos. Es importante señalar que el número de los ID diferentes de pez cebra en RefSeq o Ensembl es cerca de los 21 000 reportados recientemente en otro lenguado (*Cynoglossus semilaevis*) y no son tan diferentes de los 26 206 genes que fueron recientemente reportados en pez cebra [159, 166]. Entonces, se puede sugerir que los transcriptomas de referencia de lenguado desarrollados en este trabajo cubren la mayoría de los genes del lenguado.

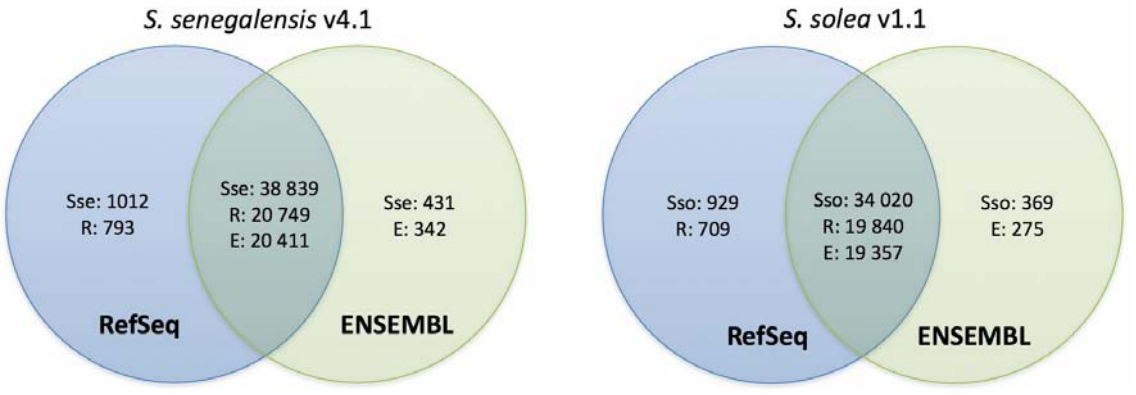

Figura IV.28: *Anotación de los transcriptomas de referencia con los ortólogos de pez cebra utilizando los ID de RefSeq y Ensembl Sse: número de transcritos de Solea senegalensis con un ID; Sso: número de transcritos de Solea solea con un ID. R: número de ID únicos de RefSeq; E: número de ID únicos de Ensembl*

IV.2.2.6. Análisis comparativo entre las dos especies de lenguado

Un análisis comparativo entre los transcriptomas de las dos especies de lenguado puede revelar nuevas pistas sobre su biología y evolución y proporcionar otros elementos para confirmar la representatividad de su transcriptoma de referencia.

S. solea y *S. senegalensis* muestran una clara similitud funcional

Para comprobar la similitud de los dos lenguados desde el punto de vista funcional, comparamos la abundancia de los términos GO de los transcritos con ortólogo de la tabla IV.14. La distribución de los términos GO por categorías entre las dos especies de lenguado reveló que eran similares (figura IV.29). El número más alto de transcritos anotados con procesos biológicos fue asociado con procesos metabólicos (15,2%) y celulares (22,2%) (figura IV.29-A). En componentes celulares, las categorías más representadas fueron células (36,3%) y orgánulos (22,1%) (figura IV.29-B). En función molecular, el mayor número de transcritos anotados fue en la categoría de actividad catalítica (30,4%) (figura IV.29-C). Curiosamente, las actividades reguladoras de canales y antioxidantes fueron representadas solo en *S. senegalensis.* La ausencia de estas dos actividades en *S. solea* seguramente se origina de la secuenciación de las muestras o del proceso del ensamblaje, sobre todo que están muy poco representadas en *S. senegalensis*, aunque no podemos descartar la existencia de una

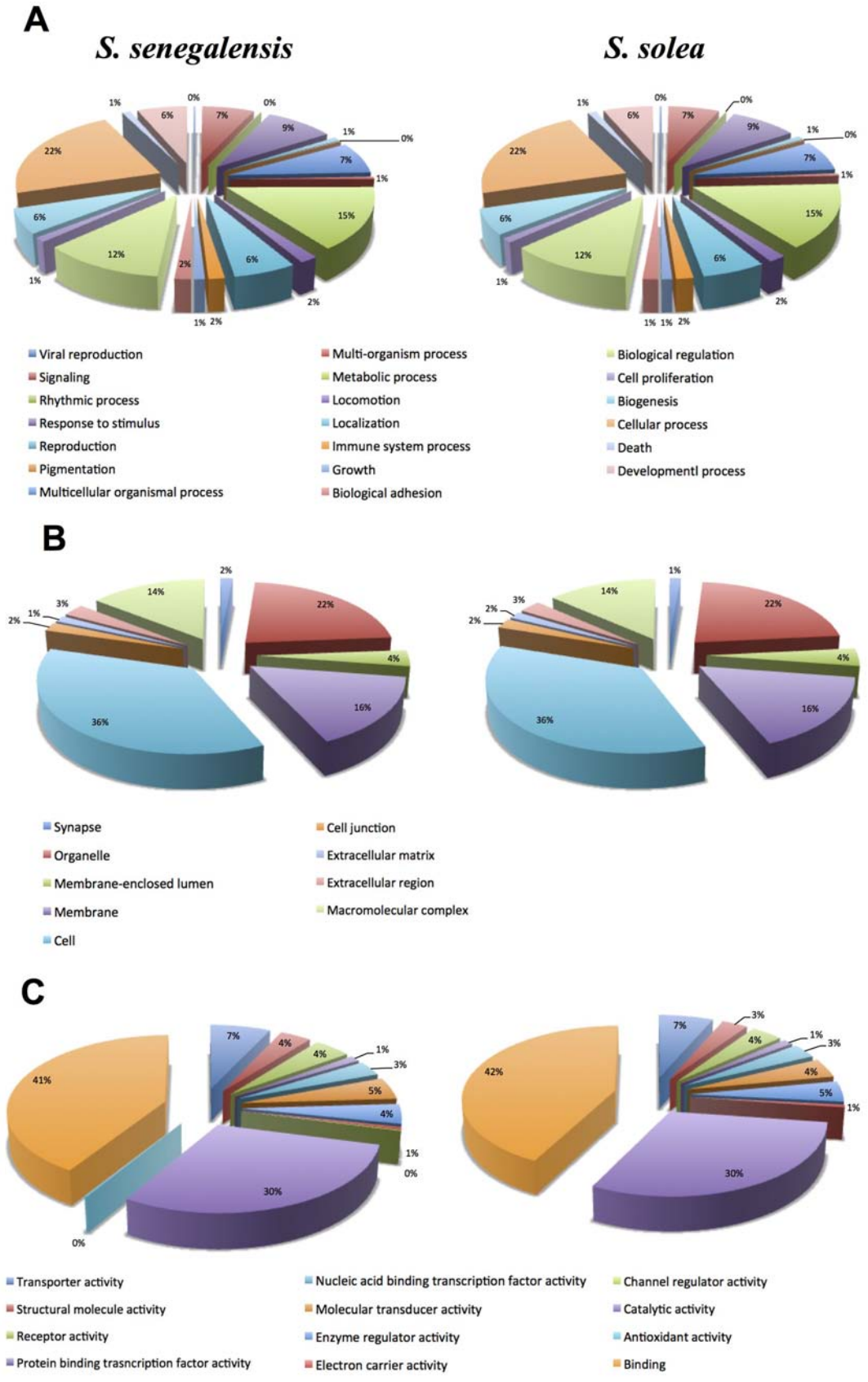

Figura IV.29: *(tomada de [143]) distribución de los Gos en los dos transcriptomas según los procesos biológicos (A), componentes celulares (B) y función molecular (C)*

razón biológica. En conclusión, los transcriptomas de lenguado son aparentemente similares del punto de vista biológico y funcional.

S. solea y *S. senegalensis* muestran una alta similitud de los genes

Otra comparación de los transcriptomas de lenguado se basó en su parecido con una tercera especie que hace de referencia, o sea, en la ortología con pez cebra. La figura IV.30 muestra que el 78,4% de los transcritos ortólogos en los lenguados con un ortólogo en pez cebra tuvieron una similitud ≥95% a nivel de nucleótido, y 1 437 de ellos eran idénticos a 100%. Como se esperaba, los transcritos sin ortología con pez cebra tuvieron un grado de identidad menor (92-96%; figura IV.30). Hay que señalar que 41 transcritos sin ortólogo tuvieron la misma secuencia y 49 transcritos mostraron 99% de identidad entre las dos especies de lenguado. Todos estos datos demuestran el alto nivel de similitud entre los dos transcriptomas de lenguado. Suponiendo que los transcritos de lenguado que comparten el mismo ortólogo de pez cebra también pueden considerarse como ortólogos de lenguado, un total de 39 851 transcritos de referencia en *S. senegalensis* y 34 949 transcritos de referencia en *S. solea* comparten 17 562 IDs para RefSeq y 17 031 IDs para Ensembl, lo cual refleja claramente el alto nivel de ortología entre estas dos especies de lenguado, y de nuevo sugiere que en el transcriptoma de referencia están incluidos los dos alelos de cada gen (al haber aproximadamente el doble de transcritos de lenguado que ID de pez cebra).

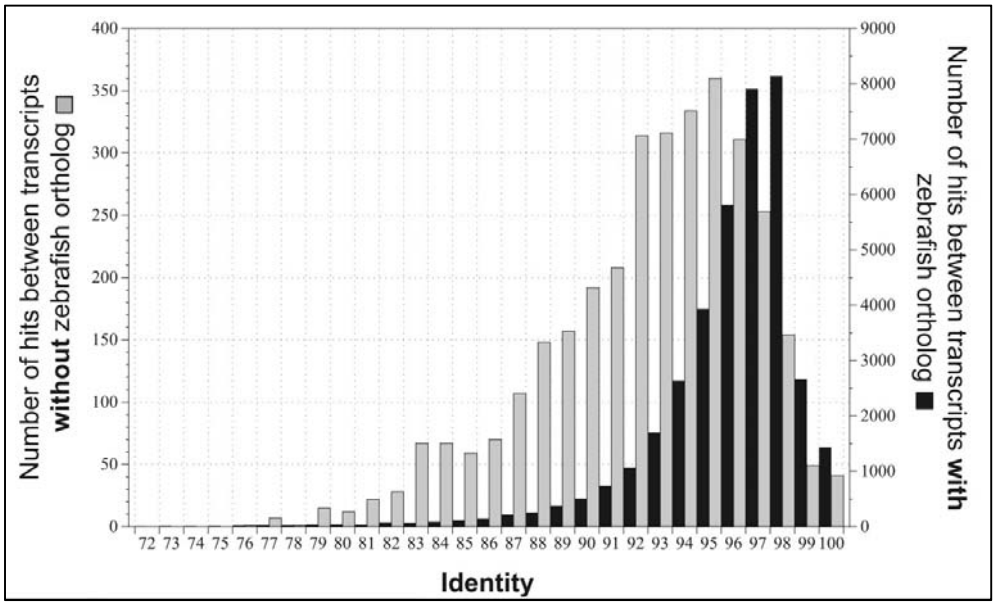

Figura IV.30: *(tomada de [143]): Distribución de los niveles de similitud entre los transcriptomas de referencia de los dos lenguados para los transcritos con (barras negras) o sin (barras gris) ortólogo de pez cebra*

S. solea y *S. senegalensis* contienen transcritos específicos del lenguado y de peces planos.

Los ortólogos verdaderos entre ambas especies de lenguado se obtuvieron después de realizar un Blast reciproco entre los transcriptomas de referencia. En este análisis, dos

transcritos fueron considerados ortólogos verdaderos cuando un alineamiento Blast reciproco y restrictivo (>97% de identidad) siempre devuelve que la misma pareja de secuencias muestra la mejor puntuación y mejor valor de *E* [84]. Un total de 26 291 transcritos de referencia de *S. senegalensis* fueron homólogos al transcriptoma de referencia de *S. solea*, y 21 238 transcritos de referencia de *S. solea* fueron homólogos al transcriptoma de referencia de *S. senegalensis*. De estas secuencias homologas solo 11 953 pueden ser considerados como verdaderos ortólogos según el criterio anterior. Estos verdaderos ortólogos incluyen 210 transcritos no anotados con una longitud media de 900 nt en *S. senegalensis* y 1 199 nt en *S. solea*. Aun más interesante, 137 de ellos tuvieron un testcode ≥ 0,94, indicando que estos probablemente codifican una proteína especializada en estos peces. Para confirmar esta hipótesis, estos transcritos se compararon con proteínas de otros peces (*Gadus morhua*, *Oryzias latipes*, *Oreochromis niloticus*, *Tetraodon nigroviridis*, *Gasterosteus aculeatus*) y solo 35 (25%) siguieron sin presentar ortología (tabla IV.15). Por otra parte, 75 transcritos (54,7%) mostraron un ortólogo claro solo en el lenguado *C. semilaevis* (el más parecido a los lenguados *Solea*), indicando que podrían ser transcritos específicos de los peces planos. En conclusión, se identificaron 11 953 transcritos que son verdaderos ortólogos entre las dos especies de lenguado, de los cuales 75 son probablemente transcritos específicos de peces planos y 35 son nuevos transcritos específicos de lenguado.

Tabla IV.15: *(tomada de [143]) Análisis de homología basado en Blast de los ortólogos de lenguado que no tienen ortología con pez cebra comparados con proteínas de referencia de otros teleósteos extraídos de Ensembl (a fecha de 1ro de Marso 2014), y secuencias genómicas de C. semilaevis de GenBank (a Fecha de 1ro Marso 2014)*

	Transcritos anotados	Transcritos no anotados
Ortólogos entre lenguados	137	351
Ortólogos en proteínas de otros teleósteos		
Gadus morhua	7	155
Oryzias latipes	10	190
Oreochromis niloticus	17	241
Tetraodon nigroviridis	6	198
Gasterosteus aculeatus	17	235
Al menos una de estas especies	27	290
Ortólogos con ADN del lenguado *C. semilaevis*	99	287
Ortólogos en teleósteos, pero no en *C. semilaevis*	3	46
Ortólogos específicos solo de *C. semilaevis*	75	43
Sin ortólogo	35	18

La comparación de los transcriptomas con teleósteos pone de manifiesto un conjunto de genes específicos de linaje

Los 11 743 verdaderos ortólogos de lenguado que tienen anotación (excluidos los 210 transcritos no anotados, más arriba) se estudiaron en función de sus ortólogos en RefSeq y/o Ensembl en el pez cebra. Como se muestra en la figura IV.31, la mayoría de los verdaderos ortólogos (93,8%) tienen un ortólogo en el pez cebra. Sin embargo, el hallazgo más interesante es el pequeño subgrupo de ortólogos de lenguado sin similitud con el pez cebra. (701 en *S. senegalensis* y 492 en *S. solea*, con 351 transcritos presentes en ambas especies; figura IV.31). Algunos de ellos se relacionaron con el sistema inmunitario, tal como péptidos antimicrobianos de hepcidina y algunas interleucinas. Estos datos muestran que este subgrupo de ortólogos de lenguado sin ortología con el pez cebra pueden representar genes de linaje específicos de los teleósteos que aparecieron más tarde en la evolución. Para comprobar la presencia de estos transcritos en otros teleósteos se compararon las proteínas deducidas de los transcritos de referencia. En la tabla IV.15 se observa que la mayoría las proteínas estaban presentes también en el genoma de *C. semilaevis* (287; 81,8%) y solo 18 transcritos (5,1%) sin ortología con los teleósteos, lo que confirma que esta colección de transcritos pueden corresponderse a genes adquiridos o fijados durante la evolución de los peces planos.

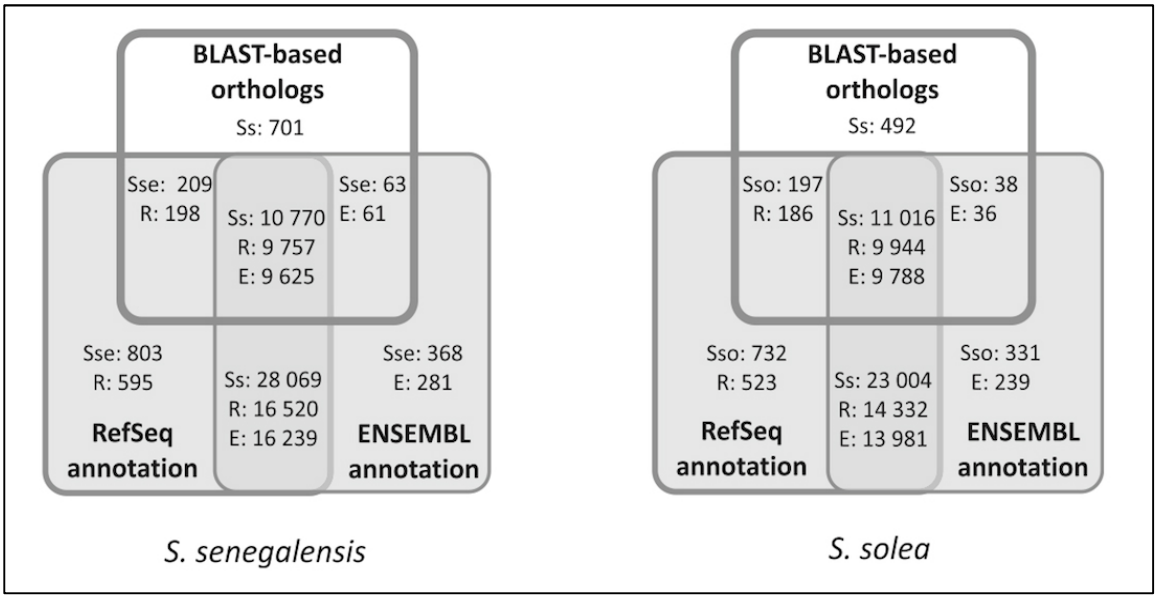

Figura IV.31: *(tomada de [143]) Diagrama de Venn que recoge las coincidencias de las especies de Solea entre los ortólogos verdaderos de lenguado y los transcritos con ortólogo RefSeq/Ensembl de pez cebra. Los diagramas comparan los 11 743 ortólogos verdaderos con los identificadores únicos RefSeq de pez cebra en SoleaDB para S. senegalensis (39 851) y S. solea (34 949) y con los identificadores únicos en Ensembl de pez cebra en SoleaDB para S. senegalensis (39 270) y S. solea (34 389). Dentro de los diagramas, los números se refieren a la cantidad de transcritos en SoleaDB para S. senegalensis (Sse) y S. solea (Sso), el número de transcritos en SoleaDB con identificador RefSeq de pez cebra (R) o con identificador Ensembl de pez cebra (E)*

IV.2.2.7. Los transcriptomas de lenguado como fuente de marcadores moleculares

La búsqueda de los SSR en el transcriptoma se realizó con MREPS (apartado III.3.4.10). Las estadísticas se calcularon a partir del fichero la salida de MREPS con un script Ruby (apartado III.3.5.6). Los SSR se determinaron en el transcriptoma global (*S. senegalensis* v4 y *S. solea* v1; tabla IV.16-A), los transcriptomas de referencia (*S. senegalensis* v4.1 y *S. solea* v1.1; (tabla IV.16-B) y el grupo de los ortólogos verdaderos (tabla IV.16-C). Conviene señalar que en los transcriptomas globales, las repeticiones dinucleotídicas fueron los SSR más abundantes, seguidos de las repeticiones tri- y tetranuleotídicas. No obstante, en los transcriptomas de referencia, la repetición más abundante fue la trinucleotídica (tabla IV.16-B). Esta diferencia en la abundancia de los motivos puede ser explicada por las restricciones selectivas impuestas por el ADN codificante de proteínas, que es más abundante en el transcriptoma de referencia. En efecto, los SSR di- y tetranucleotídicos se localizaron principalmente en las UTR (regiones no traducidas), mientras que los trinucleotídicos se localizaban en los ORF (marco abierto de lectura). Estos resultados corroboran los análisis globales del genoma donde se identificaron sesgos en la distribución de las repeticiones tri- y hexanucleotídicas en los exones que codifican proteínas de los vertebrados, invertebrados, plantas y hongos [167, 168].

Tabla IV.16 (A): *Estadísticas de los SSR para el transcriptoma global*

		S. senegalensis v4	*S. solea* v1
Transcritos con SSR		163 325	170 230
SSR totales		266 434	316 388
SSR di-nucl.	Total	107 828	126 260
	ORF	3 044	3 815
	UTR	28 353	43 489
	Desconocidos	76 431	78 956
SSR tri-nucl.	Total	96 076	114 198
	ORF	20 240	27 625
	UTR	29 074	44 139
	Desconocidos	46 762	42 434
SSR tetra-nucl.	Total	39 102	44 118
	ORF	1 321	1 921
	UTR	11 795	17 559
	Desconocidos	25 986	24 638
SSR otros	Total	23 428	31 812
	ORF	1 871	3 214
	UTR	6 503	11 946
	Desconocidos	15 054	16 652
Motivo más abundante		AC/GT (80 408)	TG (94 146)

Tabla IV.16 (B): *Estadísticas de los SSR para el transcriptoma de referencia*

		S. senegalensis v4	*S. solea* v1
Transcritos con SSR		24 352	28 670
SSR totales		49 955	67 610
SSR di-nucl.	Total	16 405	22 371
	ORF	630	884
	UTR	10 348	15 295
	Desconocidos	5 427	6 192
SSR tri-nucl.	Total	22 394	29 764
	ORF	6 301	8 523
	UTR	9 832	14 699
	Desconocidos	6 261	6 542
SSR tetra-nucl.	Total	6 935	8 829
	ORF	339	425
	UTR	4 342	6 169
	Desconocidos	2 254	2 235
SSR otros	Total	4 221	6 646
	ORF	550	885
	UTR	2 347	3 946
	Desconocidos	1 324	1.815
Motivo más abundante		AC/GT (8 965)	TG (8 220)

Tabla IV.16 (C): *Estadísticas de los SSR para los ortólogos (11 743)*

		S. senegalensis v4	*S. solea* v1
Transcritos con SSR		5 820	7 304
SSR totales		12 418	18 486
SSR di-nucl.	**Total**	3 877	5 774
	ORF	154	203
	UTR	3 679	5 507
	Desconocidos	44	64
SSR tri-nucl.	**Total**	5 953	8 818
	ORF	2 399	3 414
	UTR	3 421	5 260
	Desconocidos	133	144
SSR tetra-nucl.	**Total**	1 622	2 288
	ORF	95	113
	UTR	1 504	2 142
	Desconocidos	23	33
SSR otros	**Total**	966	1 606
	ORF	174	297
	UTR	763	1 279
	Desconocidos	29	30
Motivo más abundante		TG (886)	TG (1 350)

El motivo de SSR más frecuente fue AC/GT para dinucleótidos (74,6% en ambas especies), AGC/CCT en *S. senegalensis* y AGC/GCT en *S. solea* para trinucleótidos (21,5 y 23,1% en *S. senegalensis* y *S. solea* respectivamente), y AAAC/GTTT para tetranucleótidos (17,4 y 15,6% en *S. senegalensis* y *S. solea*, respectivamente). En el transcriptoma de referencia se observaron porcentajes similares, excepto en el hecho que AGC/GCT fue más abundante que AGC/CCT. Los motivos AC/GT resultaron ser las repeticiones SSR más frecuentes en las regiones intergénicas y los intrones de los vertebrados [167, 169]. El motivo AC/TG fue identificado como el SSR más frecuente en las secuencias largas (Roche/454) ensambladas de *S. solea* [170]. La comparación de los SSR contenidos en los ortólogos verdaderos en los lenguados identificó una serie de SSR conservados de la especie de lenguado que eran 6 596 *S. senegalensis* y 6 772 en *S. solea*, de los cuales 1 273 y 4 803 SSR en *S. senegalensis* y *S. solea*, respectivamente, se pueden considerar específicos de la especie al haberse encontrado en los ortólogos de solo una especie. Este análisis proporciona nuevos marcadores moleculares que son muy útiles para el desarrollo de los ensayos de multiplex, mapeos genéticos así como el estudio de la evolución genética en el pez plano.

En total, se identificaron 337 315 SNP en los transcriptomas de *S. senegalensis* y 381 404 en *S. solea*. Una proporción significante de SNP apareció en los transcritos con ORF (109 235 [32,4%] y 115 746 [30,3%], respectivamente) de los que aproximativamente la mitad aparecían dentro de los ORF (53 265 y 46 599, respectivamente). Estas cifras de localización de los SNP en las zonas codificantes son similares a las encontradas en otras especies de peces, que oscilan desde 17,4 a 24,7% [171-173]. Como la predicción de los SNP se basaba solo en un análisis bioinformático, todavía falta una validación empírica para eliminar los

falsos positivos y errores de secuenciación [171-174], aunque estos SNP podrían utilizarse como fuentes putativos de marcadores moleculares.

IV.2.2.8. Selección de transcritos para un estudio de microarray de oligonucleótidos a realizar sobre *S. senegalensis*

Los *microarrays* son una técnica económica para los estudios de perfiles de la expresión génica y los estudios del transcriptomas [175]. En el grupo de IFAPA Centro El Toruño (Cádiz) con el cual colaboramos, se quería realizar un estudio de *microarrays* sobre algunas funciones en *S. senegalensis* [143]. Esto requirió una preparación previa de un conjunto de transcritos con la calidad indispensable para maximizar las posibilidades de éxito del experimento. Para ello, se desarrolló un *script* en Ruby que seleccionará automáticamente los transcritos que representarán las sondas del *microarray* a partir de la anotación de Full-LengtherNext (estados de los transcritos y posiciones de los ORF), que podría extenderse a cualquier otro transcriptoma (figura IV.32). Se priorizaron primero los transcrito completos no redundantes (5 545) y luego los incompletos más largos hacia el extremo 3' (34 291). La agrupación de ambos conjuntos dio lugar a 30 119 transcritos diferentes. Para complementar el conjunto de sondas, se añadieron 21 099 transcritos codificantes que se seleccionaron en función de la concordancia entre el resultado de los dos programas Testcode y ORF-Predictor. En total, se seleccionaron 51 218 transcritos clasificados por orden de calidad. El grupo de Manuel Manchado (IFAPA EL Toruño) mandó imprimir estas sondas y realizó con éxito un análisis del efecto de la salinidad sobre el desarrollo de las larvas de *Solea senegalensis* con el *microarray* (resultados no mostrados).

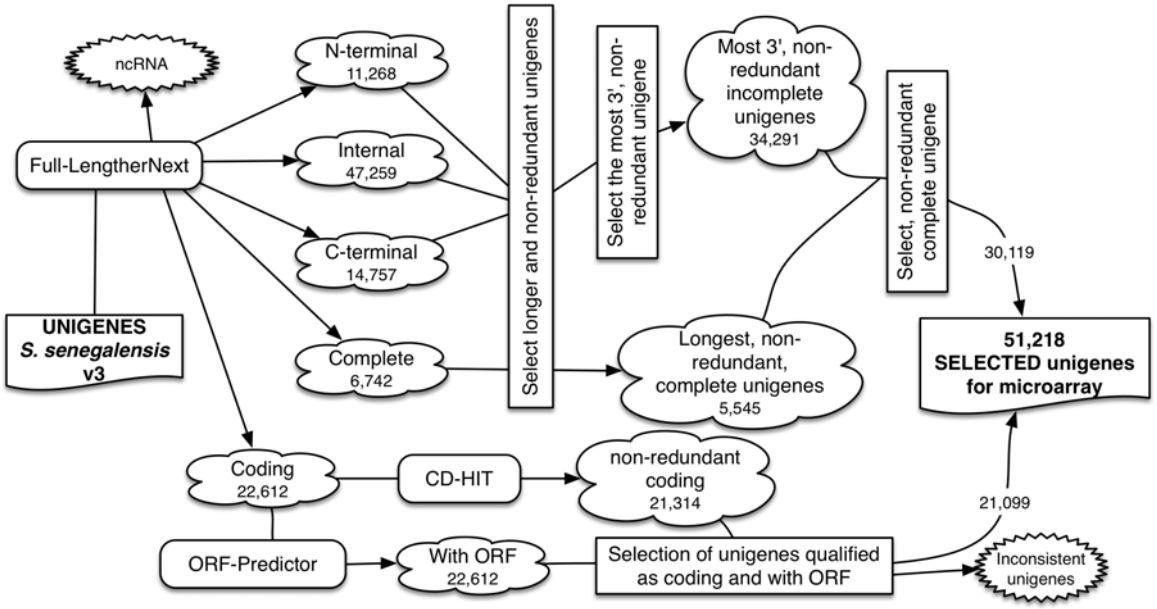

Figura IV.32: *(tomada de [143]) Representación esquemática de la estrategia de selección de las sondas para la construcción del microarray de oligonucleótidos para el lenguado senegalés*

IV.2.3. Transcriptoma de la microalga *Tisochrysis lutea*

El potencial económico de las microalgas está en crecimiento continuo [176]. Sus aplicaciones son múltiples y van desde la purificación de las aguas hasta la alimentación animal y humana [177]. *Isochrysis galbana* recientemente llamada *Tisochrysis lutea* (Tiso) [178] es una microalga muy utilizada en el sector de acuacultura como fuente de proteínas y vitaminas. Por eso se utiliza en la dieta alimentaria de las larvas y la fase juvenil de los peces. Estudios sobre el transcriptoma permitirían ampliar la investigación sobre varios aspectos como el metabolismo y el ciclo de vida de esta especie.

IV.2.3.1. Preprocesamiento de las librerías

Las muestras de *Tisochrysis lutea* secuenciadas se prepararon en el laboratorio del grupo de investigación de IFAPA El Toruño (Cádiz). El resumen del preprocesamiento con SeqTrimNext de las lecturas normalizadas y sin normalizar obtenidas con las tecnologías Roche/454 e Illumina (figura IV.33) se presenta en la tabla IV.17. Los contaminantes más representados fueron el plastidio, el RNA ribosómico y otros microorganismos, especialmente hongos como *Aspergillus niger* y *Schizosaccharomyces pombe.* El porcentaje de lecturas no normalizadas útiles fue alto para Roche/454 y muy alto para Illumina (69,9% y 93,0% respectivamente) mientras que en las lecturas normalizadas estos porcentajes fueron bastante bajos tanto para Roche/454 como para Illumina (32,4% y 31,8% respectivamente). Estos bajos porcentajes de lecturas normalizadas útiles se deben al exceso de secuencias contaminadas (52,3% en Roche/454 y 51% en Illumina) de las cuales, la mayor parte fueron lecturas del plastidio.

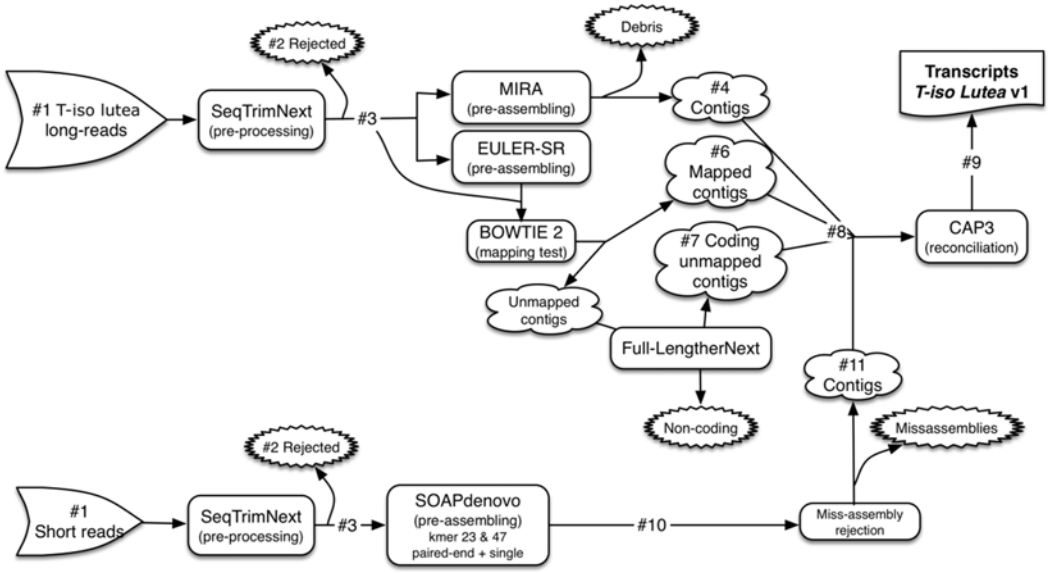

Figura IV.33: *Estrategia de pre-procesamiento, ensamblaje y la reconciliación del transcriptoma de Tisochrysis lutea*

Tabla IV.17: *Resumen del preprocesamiento de las lecturas originales del transcriptoma de T. lutea*

	Referencia a la figura IV.33	Roche/454	Illumina
Total lecturas	#1		
Normalizadas		1 151 067	925 682 496
No normalizadas		505 781	612 903 376
Longitud media			
Normalizadas		547	100
No normalizadas			76
Lecturas rechazadas	#2		
Normalizadas		602 104 (66,2%)	628 141 506 (68%)
No normalizadas		48 828 (25,7%)	34 565 829 (5,6%)
Contaminación			
Normalizadas		602 104 (52,3%)	472 474 387 (51%)
No normalizadas		48 828 (9,7%)	21 926 541 (3,6%)
Total de lecturas útiles	#3		
Normalizadas		373 149 (32,4%)	294 921 874 (31,8%)
No normalizadas		353 784 (69,9%)	569 737 072 (93,0%)
Lecturas pareadas			
Normalizadas		-	205 762 484 (22,2%)
No normalizadas		-	555 513 714 (90,9%)
Lecturas simples			
Normalizadas		-	89 159 390 (9,6%)
No normalizadas		-	14 223 358 (2,3%)
Longitud media			
Normalizadas		233	84
No normalizadas			67

IV.2.3.2. Ensamblaje

El transcriptoma de *T. lutea* fue inicialmente ensamblado siendo el mismo protocolo utilizado para el transcriptoma de *Solea senegalensis* (apartado IV.2.2.2) donde se emplearon las lecturas Roche/454 y las lecturas Illumina normalizadas. Con esto se obtuvo la versión 1 del transcriptoma de la microalga. Para obtener más cobertura del transcriptoma y más genes completos, se realizó otro ensamblaje donde se añadieron las lecturas no normalizadas (RNA-seq). En este segundo ensamblaje se aplicó el mismo protocolo utilizado para ensamblar el transcriptoma de *S. senegalensis* (Apartado IV.2.2.2), excepto en la parte del ensamblaje de las lecturas cortas donde Oases fue reemplazado por SOAPdenovo-Trans [179] ya que este era más rápido para ensamblar grandes cantidades de lecturas [161], como en este caso. De hecho SOAPdenovo-Trans fue capaz de ensamblar todas las librerías Illumina (normalizadas y no normalizadas) juntas en menos de 1 día, lo cual no fue posible con Oases utilizando el k-mero 23 debido a que el proceso de ensamblaje fue interrumpido al consumir el tiempo máximo asignado a los procesos encolados (una semana) (figura IV.33). En la tabla IV.18 se muestra el número de secuencias en cada etapa del protocolo del ensamblaje así como las características del transcriptoma generado. El número total de transcritos fue 149 517 con una longitud media de 499.

Tabla IV.18: *Número de secuencias en cada etapa del ensamblaje de T. lutea y las características de la versión 1 del transcriptoma*

			Referencia a la figura IV.33	
454	Contigs de MIRA		#4	27 668
	Contigs de EULER	Mapeados	#6	12 067
		No mapeados codificantes	#7	71
	Contigs totales 454		#8	39 806
Illumina	Contigs de SOAPdenovo		#10	191 682
	Secuencias útiles		#11	190 473
	Reconciliación CAP3		#9	149 517

Limpieza adicional aplicada sobre de los transcritos.

Aunque las muestras secuenciadas procedían de un cultivo homogéneo (poca variabilidad genética), el número de transcritos fue muy superior al número de proteínas de la haptofita más cercana *Emiliania huxleyi* (39 125). Esto nos hizo sospechar que los transcritos podrían incluir secuencias que no eran de *T. lutea*. Por lo tanto, realizamos una nueva búsqueda de contaminantes que consistió, en un primer lugar, en alinear los transcritos contra el genoma humano ya que este por su tamaño muy grande no estaba incluido en la base de datos de contaminantes de SeqTrimNext. Un total de 37 130 secuencias se alinearon con el genoma de humano y por lo tanto fueron descartadas. En un segundo lugar, el resto de los transcritos fueron comparados contra la colección de nucleótidos (*Nucleotide collection nr/nt*) de NCBI utilizando blastn (`http://blast.ncbi.nlm.nih.gov/Blast.cgi`), lo cual permitió descartar otras 42 082 secuencias de las cuales algunas fueron de la bacteria *Kordia algicida,* pero la mayoría fueron de plantas. Después de esta segunda "limpieza" de contaminantes que, según nos constó, procedía de las librerías de Illumina sin normalizar, el número final de transcritos se quedó en 70 306, los cuales formaron la versión 2 del transcriptoma. En la tabla IV.19 se muestra una comparación entre las dos versiones 1 y 2 de *T. lueta.* Se nota que aunque el número de transcritos se redujo a más de la mitad, el número de transcritos con tamaño superiores a 500 nt permaneció casi invariable, por lo que la mayoría de los transcritos eliminados como contaminantes desde la versión 1 en su totalidad eran pequeños (<500 nt). La eliminación de estos elementos contaminantes hizo que la longitud media de los transcritos pasara de 499 pb a 904 pb y el N50 de 1447 a 1646.

Tabla IV.19: *Características de las dos versiones del transcriptoma de T. lutea*

	T. lutea v1	*T. lutea* v2
Transcritos	149 517	70 306
Nº transcritos > 500 nt	37 417	36 385
Longitud media	499	904
N50	1447	1646
Transcrito más largo	10 309	10 309

IV.2.3.3. Anotación del transcriptoma e inclusión en base de datos

En la anotación del transcriptoma de *T. lutea* (v1 y v2) se aplicó el mismo flujo de trabajo utilizado en *Solea senegalensis* (apartado IV.2.2.3). Los datos de anotación resultantes fueron incluidos en la base de datos IsochrysisDB (`http://www.scbi.uma.es/isochrysisdb/`) cuya portada se muestra en la figura IV.34. La estructura y la interfaz de la base de datos se basó en SoleaDB, la de *Solea senegalensis* (apartado IV.2.2.4).

Figura IV.34: *Portada de la base de datos IsochrysisDB*

IV.2.3.4. Análisis del transcriptoma de *Tisochrysis lutea*

En la figura IV.35 se muestra la distribución de longitudes de los transcritos en la versión 2 del transcriptoma. Se nota una abundancia de los transcritos con longitudes entre 100 y 300 nt, En el margen de longitudes superiores a 300 y hasta 2 000 nt, el número de transcritos fue bastante uniforme.

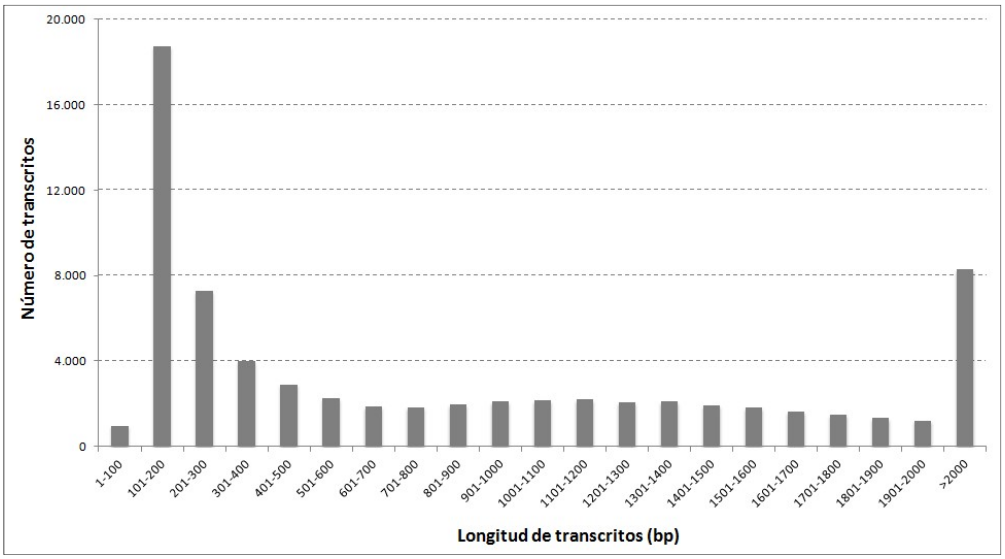

Figura IV.35: *distribución de las longitudes de los transcritos de T. lutea v2*

En la tabla IV.20 se muestra el resumen del análisis con Full-LengtherNext del transcriptoma de *T. lutea* v2. Se nota que la gran mayoría de transcritos fueron validos, ya que los artefactos formaron solo 4,7% del transcriptoma. El porcentaje de los transcritos anotados con un ID de ortólogo fue 34%, de estos IDs de ortólogos 47,6% fueron únicos y 23,2% fueron completos únicos. La parte más grande de los transcritos fueron sin ortólogo (65,2%), aunque una buena parte de ellos (42,9%) fueron codificantes por lo que representaron los nuevos transcritos o bien formas inmaduras y truncadas de otros transcritos conocidos (resultados no mostrados). Después de la eliminación los artefactos y la resolución de las quimeras, el transcriptoma global se quedó con 69 800 transcritos. De estos, Full-LengtherNext seleccionó 23 671 transcritos que formaron el transcriptoma representativo.

Tabla IV.20: *Características del transcriptoma ensamblado de T. lutea v2*

	Transcritos	%
Nº de transcritos	70 306	
Artefactos	3 339	4,7
Transcritos validos	70 306	100
>500pb	35 554	50,6
>200pb	50 035	71,2
Transcrito más largo	10 309	-
Transcritos con ortólogo	23 939	34,0
Diferentes ortólogos IDs	11 392	47,6
ORFs completos	9 199	38,4
ORFs completos diferentes	5 546	23,2
C-terminus	6 661	27,8
N-terminus	2 764	11,5
Internal	5 315	22,2
ARNs no codificantes putativos	27	0,0
Transcritos sin ortólogo	45 834	65,2
Nuevos transcritos putativos	19 645	42,9
Desconocidos	26 189	57,1
Transcriptoma de referencia	23 671	-

A partir del transcriptoma global (versión 2), 6 889 secuencias fueron anotadas con al menos un GO. El análisis de las funciones procedentes de la anotación con los términos GO (figura IV.36) muestra que el número más alto de transcritos anotados con procesos biológicos fue asociado con procesos celulares (26%) y metabólicos (24%) (figura IV.36-A). En lo que se refiere a los componentes celulares, las categorías más representadas fueron células (35%) y orgánulos delimitados con membranas (19%) (figura IV.36-B). Respecto a la función molecular, el mayor número de transcritos anotados fue en la categoría de actividad catalítica (52%) y fijación a receptores (36%) (figura IV.36-C). Por otro lado, 790 transcritos fueron asignados a un mapa de KEGG universal. Globalmente, las enzimas que intervienen en las rutas más importantes como aquellas del metabolismo del nitrógeno, de la fotosíntesis y del metabolismo de azúcar (figura IV.37) estaban bien representadas en los mapas de KEGG

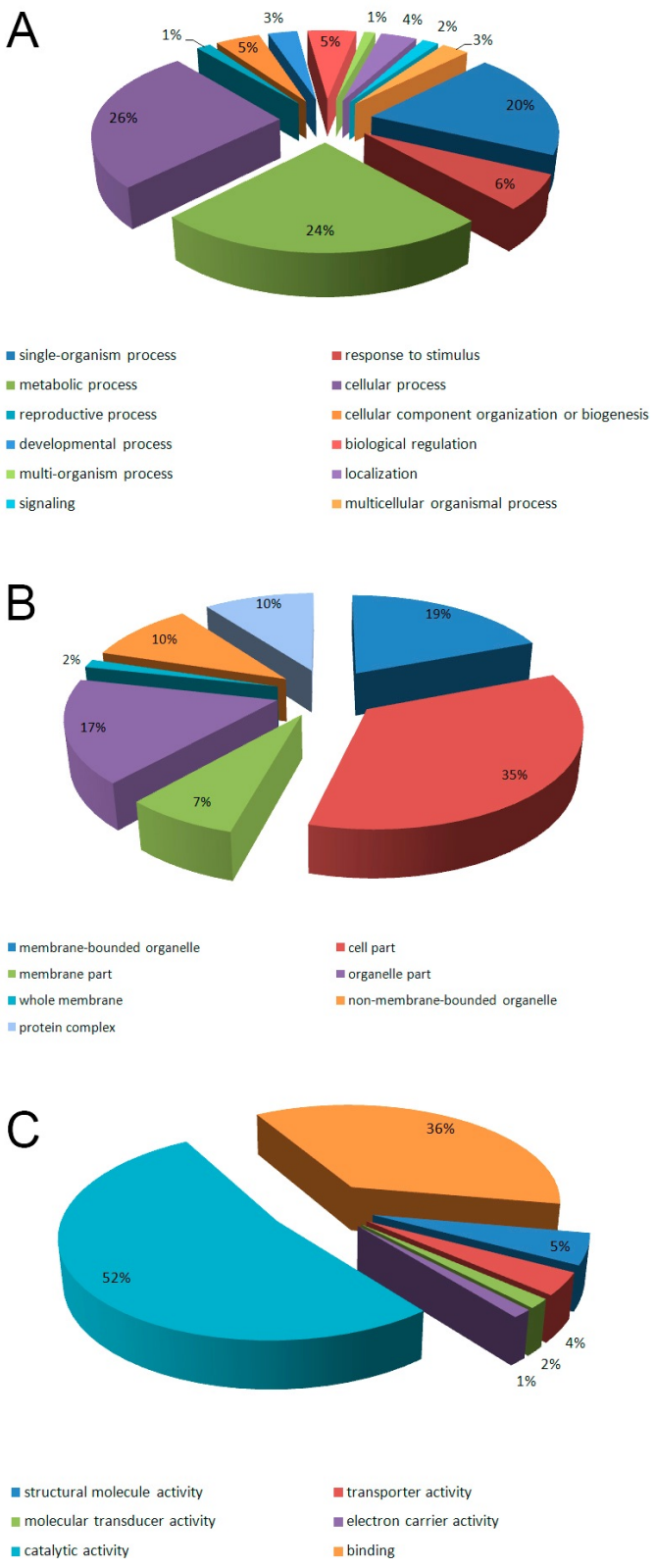

Figura IV.36 : *Distribución de los GO del transcriptoma de T. lutea según los procesos biológicos (A), componentes celulares (B) y función molecular (C)*

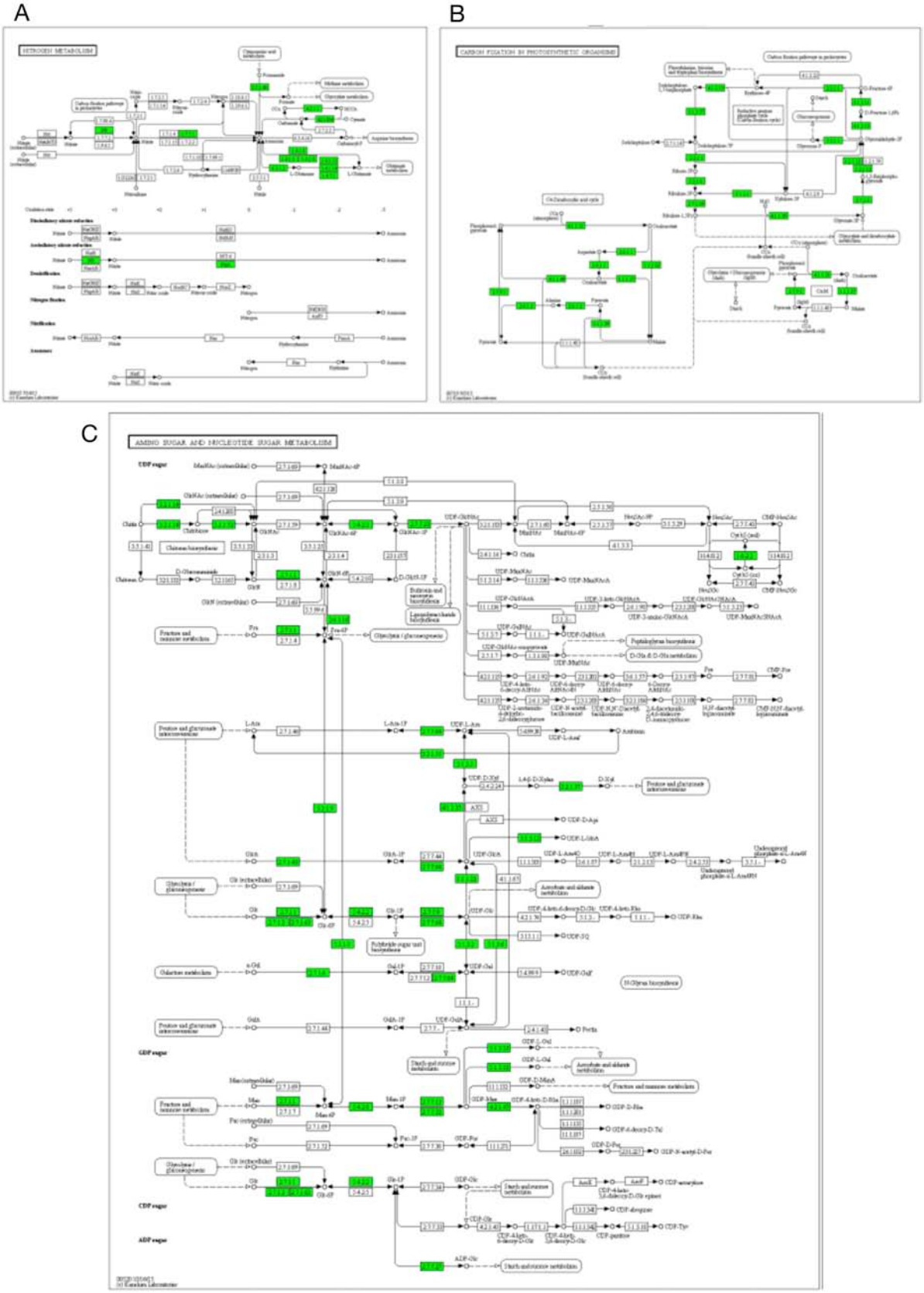

Figura IV.37: *Rutas metabólicas relacionadas con el metabolismo del nitrógeno (A), fijación de carbono (B) y el metabolismo de azúcar (C) según KEGG, en donde aparecen las enzimas de todos los organismos. Se resaltan en verde las enzimas de T. lutea*

Marcadores SSR contenidos en el transcriptoma de *T. lutea*

Por la importancia que tienen los marcadores moleculares en la exploración del genoma y en la mejora genética, en la tabla IV.21 se presentan algunas estadistas sobre los marcadores SSR encontrados en la versión 2 del transcriptoma global y en el de referencia. Se observa que en ambos casos las repeticiones trinucleotídicas fueron los SSR más abundantes seguidas con las repeticiones dinucleotídicas y superiores a tetra-nucleótidos («SSR, otros»).

Tabla IV.21: *Estadísticas de los SSR para el transcriptoma global y de referencia de T. lutea*

	Transcriptoma global	Transcriptoma de referencia
Transcritos con algún SSR	9 500	4 650
SSR totales	12 277	5 937
SSR di-nucleotídicos	2 677	1 068
SSR tri- nucleotídicos	5 280	2 865
SSR tetra- nucleotídicos	1 763	8 40
SSR, otros	2 557	1 164
Motivo más abundante	GA (644)	GA (254)

Comparación de *T. lueta* con otras especies.

El transcriptoma de referencia de *T. lutea* v2 fue comparado con las proteínas de 6 microalgas cercanas (2 clorófitas, 2 glaucofitas, una rodófitas y una haptofita). El número de proteínas para cada microalga y el número de les genes homólogos entre *T. lutea* y estas microalgas de referencia se muestran en la figura IV.38. Como se esperaba, al ser *Emiliana huxleyi* una especie del mismo linaje y muy cercana de *T. lutea,* tiene la proporción más alta de genes homólogos con *T. lutea* (43%). Esta proporción se redujo al 18% con las microalgas de referencia de otros linajes. Por otra parte, el transcriptoma de referencia de *T. lutea* fue comparado con las proteínas RefSeq y Ensembl de *E. huxleyi.* (figura IV.39), mostrando que 10 216 transcritos fueron ortólogos a 8 170 entradas RefSeq y 8 109 entradas de Ensembl. Muchas haplofitas, incluida *E. huxleyi,* se sabe que tienen un ciclo de vida haplodiplonte en el cual ambas fases haploide y diploide son capaces de una reproducción asexual [180]. En *T. lutea*, el nivel de ploidía no está descrito [181]. En cultivo, los *Isochrysidaceae* tienen un único morfotipo sin calcificación y se cree que la calcificación de la fase diploide se implicó solo una vez en el origen de los cocolitóforos y aparentemente desapareció temprano en la historia evolutiva de los *Isochrysidaceae* [182]. Así, los genes encontrados en *T. lutea* son probablemente de la etapa haploide aunque no podemos ser seguros de ello. La realización de más estudios sobre el ciclo de vida de esta especie pueden ofrecer más conocimientos sobre la interacción de los alelos y la función de sus genes.

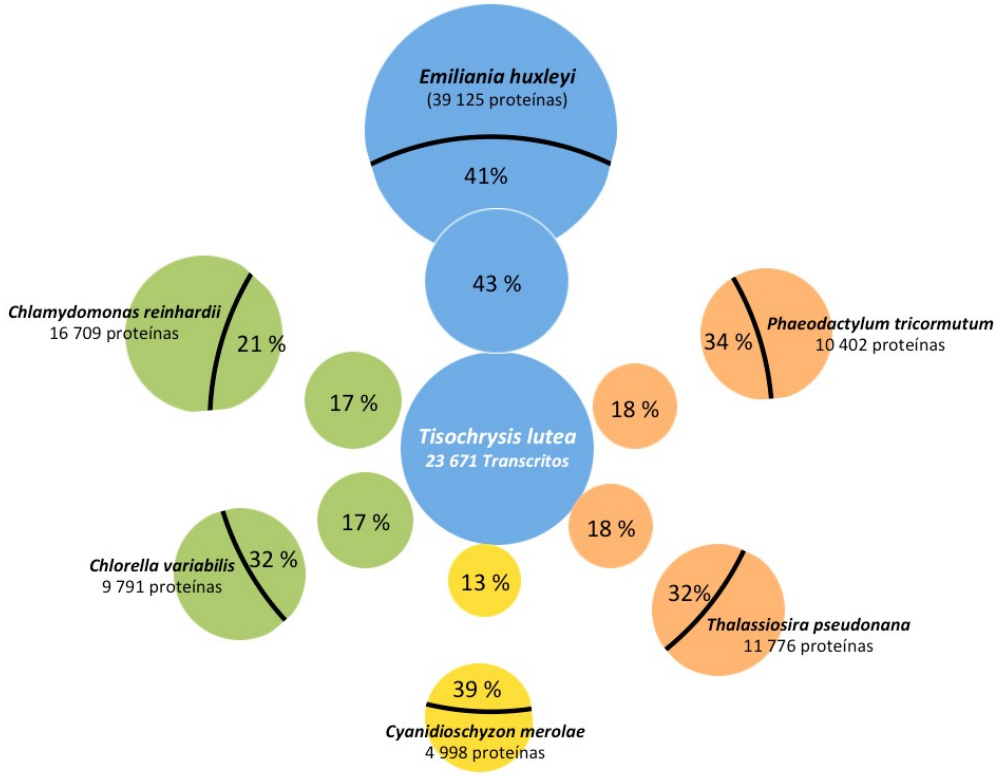

Figura IV.38: *Análisis comparativo del transcriptoma de referencia de T. lutea con las proteínas de microalgas de referencia. Los tamaños de los círculos son relativos al número de transcritos o proteínas. Códigos de colores: verde: clorófitas, amarillo: rodófitas, naranja: glaucofitas, azul: haptofitas. Los números en los círculos periféricos forman la proporción (%) de los genes de los organismos de referencia homólogos con T. lutea. Los números en los círculos interiores forman la proporción (%) de los genes de T. lutea homólogos con los organismos de referencia*

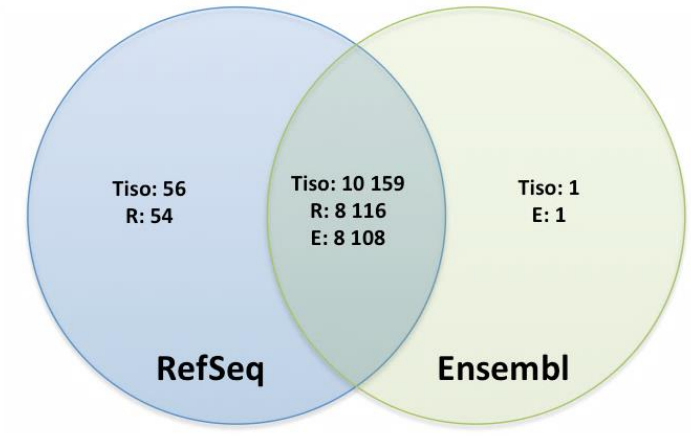

Figura IV.39*: Anotación del transcriptma de referencia con los ortólogos de Emiliania huxleyi utilizando los IDs de RefSeq y Ensembl*

IV.2.4. Transcriptoma de *Ruditapes decussatus* (almeja fina)

La almeja fina, también conocida como almeja de carril es un bivalvo filtrador. Es habitante común de la zona intermareal y submareal somera de fondos blandos en estuarios y costas del norte de España donde forma un recurso económico importante [183]. Los principales inconvenientes de la almeja fina para el cultivo son una tasa de crecimiento lenta y la susceptibilidad a la perkinsosis, una enfermedad parasitaria que puede provocar gran mortalidad. Existe el convencimiento de que estos inconvenientes podrían ser remediados con programas de mejora genética adecuados, para lo que cualquier información sobre la genómica de esta especie sería de gran ayuda.

En este capítulo vamos a aprovechar de los flujos de ensamblaje y anotación desarrollados anteriormente para ensamblar y anotar el transcriptoma de la almeja fina.

IV.2.4.1. Preprocesamiento de las librerías

Las muestras de *R. decussatus* secuenciadas fueron preparadas en el laboratorio del grupo de investigación de IFAPA El Toruño (Cádiz). Las lecturas brutas se sometieron a un preprocesamiento con SeqTrimNext (figura IV.40) cuyo resultados se muestran en la tabla IV.22. Los contaminantes más importantes fueron la mitocondria, el RNA ribosómico y algunos hongos como *Penicillium chrysogenum* y *Aspergillus niger*. Las lecturas rechazadas formaron un porcentaje pequeño de las lecturas brutas (8,9%) porque se recuperaron la mayoría de éstas como lecturas útiles (91,1%).

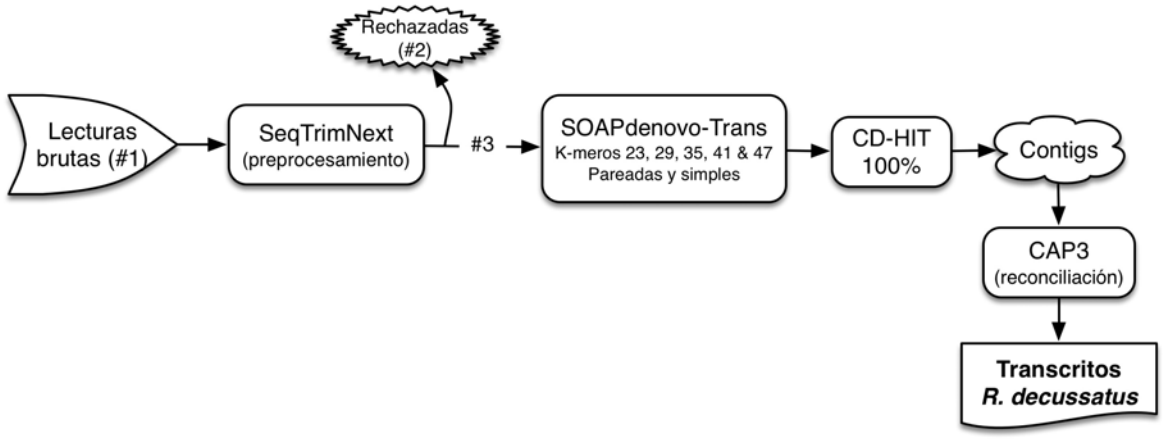

Figura IV.40: Estrategia de preprocesamiento, ensamblaje y reconciliación del transcriptoma de *R. decussatus*

Tabla IV.22: *Resumen del pre-procesamiento de las lecturas originales de R. decussatus*

	Referencia a la figura IV.40	Illumina	%
Total lecturas	#1	127 327 692	
Longitud media		76	-
Lecturas rechazadas	#2	11 351 692	8,9
Contaminación		9 335 020	7,33
Total de lecturas útiles	#3	115 954 012	91,1
Lecturas pareadas		112 363 586	88,3
Lecturas simples		3 590 426	2,82
Longitud media		68	-

IV.2.4.2. Ensamblaje

En este ensamblaje, debido a que las lecturas de entrada procedían de una sola librería y a que su número era bajo, nos hizo decantar por el uso de 5 diferentes k-meros (23, 29, 35, 41 y 47) a fin de obtener un transcriptoma con la máxima cobertura de los genes. Se probaron dos ensamblajes del transcriptoma: uno con Oases (que se utilizó para ensamblar el transcriptoma de *S. senegalenses*; apartado IV.2.2.2) y otro de SOAPdenovo-Trans (que se utilizó para ensamblar el transcriptoma de *T. lutea*; apartado IV.2.3.2) para seleccionar el mejor de ellos. En la tabla IV.23 se muestran algunos datos que reflejan la calidad del transcriptoma generado por los dos ensambladores. Tal como se observa, el transcriptoma generado por SOAPdenovo-Trans en comparación con el generado por Oases tuvo más transcritos (328 000 y 225 281 respectivamente), más ortólogos únicos (39 816 y 26 026 respectivamente) y más regiones codificantes completas diferentes (4 156 y 3 975 respectivamente). Por lo tanto, seleccionamos el ensamblaje generado por SOAPdenovo-Trans (estrategia utilizada ilustrada en la figura IV.40) por haber proporcionado más variabilidad y más completitud de genes, y con un porcentaje de quimeras casi insignificante (0,6%), aunque sea más alto que el de Oases.

En comparación con Oases, SOAPdenovo-Trans consiguió proporcionar más cobertura de los transcritos y reconstruir más transcritos completos, además, tal como hemos visto en el ensamblaje de *Tisochrysis lutea* (apartado IV.2.3.2), SOAPdenovo-Trans es mucho más rápido. Por lo cual, este programa puede constituir una buena opción para realizar ensamblajes de transcriptomas.

***Tabla IV.23**: Ensamblaje de R. decussatus con Oases y con SOAPdenovo-Trans*

	Ensamblador	
	Oases	**SOAPdenovo-Trans**
Scaffolds	225 281	328 000
Media de longitudes	579	670
Ensamblajes erróneos	6	10
Quimeras	537	1 894
Ortólogos	26 026	39 816
Ortólogos diferentes	11 116	12 730
Completos	5 898	8 730
Completos diferentes	3 975	4 156

IV.2.4.3. Anotación e inclusión en base de datos

Para anotar el transcriptoma de *R. decussatus* se aplicó el mismo flujo de trabajo utilizado en *S. senegalensis* (apartado IV.2.2.3). Los datos de anotación resultantes fueron incluidos en base de datos RuditapesDB (`http://www.scbi.uma.es/ruditapesdb/`) cuya portada se muestra en la figura IV.41. La cual se basó sobre la misma estructura de la base de datos SoleaDB de *Solea senegalensis* (apartado IV.2.2.4)

***Figura IV.41:** Portada de la base de datos RuditapesDB*

IV.2.4.4. Análisis del transcriptoma de *R. decussatus*

La distribución de longitudes de los transcritos ilustrada en la figura IV.42 muestra que los transcritos largos se encuentran poco representados en el transcriptoma mientras que los transcritos cortos (100-200 nt) forman una parte grande (casi la mitad) del transcriptoma, lo cual se puede explicar por el hecho de que las lecturas ensambladas eran cortas (70 nt) y que su cobertura no era suficientemente alta para compensar su pequeño tamaño.

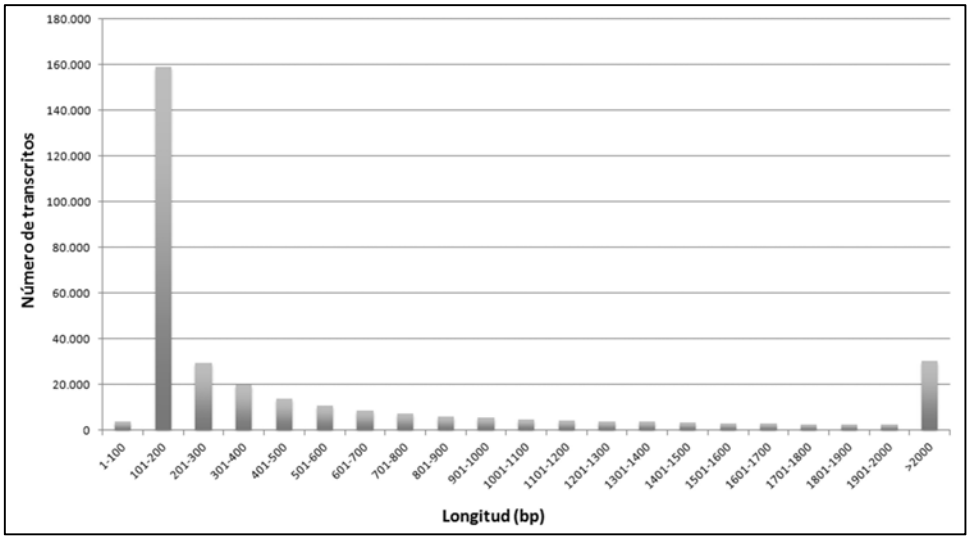

Figura IV.42: *Distribución de las longitudes de los transcritos de R. decussatus*

El análisis del resultado con Full-LengtherNext (tabla IV.24) muestra que, a pesar de que el número de transcritos es alto, los artefactos constituyen un porcentaje insignificante del transcriptoma (0,5%), de hecho el 99,5% de transcritos fueron válidos. No obstante, el porcentaje de los transcritos anotados con un ID de ortólogo fue bastante bajo (12,2 %). De estos IDs de ortólogo, el 32% fueron únicos y el 14% fueron completos únicos. Esta baja tasa de anotación puede ser explicada por la fragmentación del transcriptoma, ya que muchas secuencias por su tamaño pequeño no pasaron los filtros de anotación. En efecto, la mayoría de los transcritos, el 87,8%, no tenían ID de ortólogo aunque una parte de ellos (13,9 %) fueron cualificados como codificantes.

Tabla IV.24: *Características del transcriptoma ensamblado de R. decussatus*

	Transcritos	%
Nº de transcritos	328 000	-
Artefactos	1 690	0,5
Transcritos validos	328 342	99,5
>500pb	165 548	50,5
>200pb	101 943	31
Transcrito más largo	21 754	-
Transcritos con ortólogo	39 909	12,2
Diferentes ortólogos IDs	12 764	32
ORFs completos	14 071	35,3
ORFs completos diferentes	5 596	14,0
C-terminus	6 202	15,5
N-terminus	4 012	10
Internal	15 624	39,1
ARNs no codificantes putativos	10	0,0
Transcritos sin ortólogo	288 423	87,8
Nuevos transcritos putativos	39 980	13,9
Desconocidos	248 443	86,1

El número alto de transcritos indica que el transcriptoma ensamblado de *R. decussatus* está claramente sobreestimado lo cual probablemente provenga de la existencia de genes parálogos, formas de corte y empalme o del alto grado de fragmentación. De la tabla anterior, podemos deducir que este organismo parece contener unos 12 764 genes conocidos (de los que casi la mitad [5 596] contienen el marco abierto de lectura completo). Es curioso que este valor sea mayor que el número de entradas del transcriptoma de una almeja cercana, *Ruditapes philippinarum,* asociadas con una proteína o transcrito (9 747 entradas) [184]. Por lo tanto, el transcriptoma ensamblado incluye seguramente varios alelos para cada gen, además de muchos genes completos, por lo que puede formar una buena base desde la cual se puede seleccionar un conjunto de genes representativos y realizar análisis más profundos.

IV.3. Ensamblajes de genoma

IV.3.1- Genoma de *Photobacterium damselae* subsp. *piscicida*

Photobacterium damselae subsp. *piscicida* (conocida anteriormente como *Pasteurella piscicida*) es una bacteria gramnegativa con forma de bastón que provoca la pasteurelosis, una enfermedad que causa pérdidas enormes en la producción mundial de la acuicultura [185]. En 1999 se describió la primera pasteurollosis que afectó al lenguado senegalés cultivado en piscifactoría en el suroeste de España [186]. Desde entonces, todas las piscifactorías de lenguado en España sufrieron mortalidades serias causadas por *P. damselae* subsp. *piscicida*, después de lo cual ésta bacteria se transformó en una gran amenaza para el

cultivo del lenguado senegalés. Los estudios del genoma *P. damselae* subsp. *piscicida* pueden aportar conocimientos sobre sus factores de patogenecidad y virulencia, y abrir nuevas vías para planificar estrategias de lucha contra la pasteurelosis.

En este apartado se presentarán los ensamblajes del genoma de dos cepas de la bacteria *P. damselae* subsp. *piscicida*. La primera cepa (L091106-03H) procede de brotes de pasteurelosis en lenguado, aislados en el laboratorio del grupo de investigación de IFAPA El Toruño (Cádiz) y la segunda (DI21) se aisló a partir del hígado de peces con pasteurelosis en el laboratorio del departamento de Microbiología e Parasitología de la Universidad de Santiago de Compostela. La cepa DI21 ya había sido secuenciada y tenía borrador del genoma con el número de acceso GCA_000300355.3. Decidimos pues realizar el ensamblaje de las dos cepas con el mismo procedimiento.

IV.3.1.1. Preprocesamiento de las lecturas brutas

Como la secuenciación del ADN de las bacterias se hizo sobre una extracción de DNA total de las células, era posible que las librerías de lecturas, además del cromosoma, incluyeran secuencias de los plásmidos de estas cepas (M. Lemos y J.J. Borrego, comunicación personal). Así, para descartar estas secuencias, incluimos en la base de datos de SeqTrimNext las secuencias de los 3 plasmidos de DI21 [187] que fueron las siguientes: pPHDP10 (NC_013775.1), pPHDP60 (NC_020277.1) y pPHDP70 (KP100338.1).

Los resultados del procesamiento con SeqTrimNext (figuras IV.43 y IV.44) de las lecturas brutas de las dos cepas se presentan resumidos en la tabla IV.25. En lo que se refiere a L091106-3H, SeqTrimNext rechazó el 32% de las lecturas pareadas y el 16,6% de las lecturas simples. Respecto a DI21, el porcentaje de lecturas rechazadas fue más alto (55% de las pareadas y 28% de las simples), con lo que el número total de lecturas útiles en ambas cepas acabó resultado ser similar: de L091106-03H se recuperaron 382 755 lecturas útiles, de las cuales 69 318 (18,1%) fueron pareadas y 313 437 (81,9%) fueron simples, mientras que de DI21 se recuperaron 396 450 lecturas útiles, de las cuales 132 550 (33,4%) fueron pareadas y 263 900 (66,6%) fueron simples.

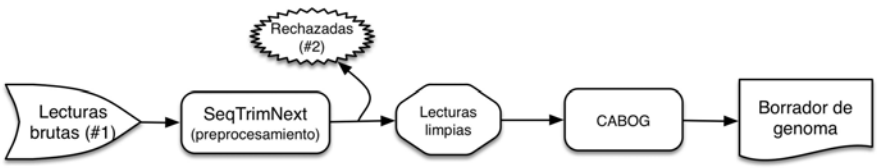

***Figura IV.43**: Estrategia de preprocesamiento, ensamblaje y scaffolding de las lecturas de la versión 1 de L091106-03H*

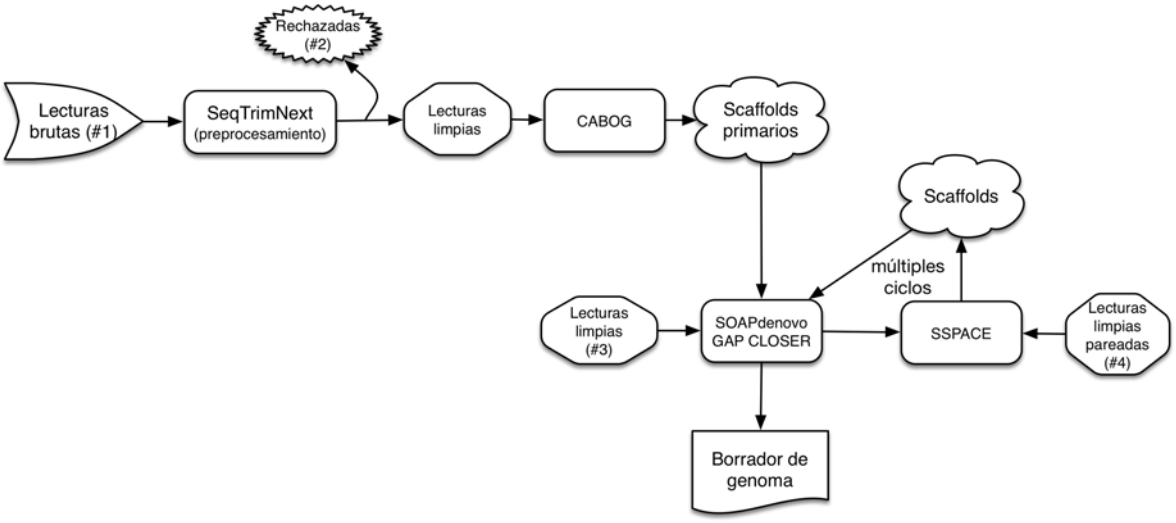

***Figura IV.44**: Estrategia de preprocesamiento, ensamblaje y scaffolding de las lecturas de L091106-03H (v2) y DI21*

***Tabla IV.25:** Resumen del pre-procesamiento de las lecturas originales de L091106-03H y DI21*

		Cepas	
	Referencia a las figuras IV.43 y IV.44	**L091106-03H**	**DI21**
Total lecturas	#1		
Pareadas		148 622	433 717
Simples		297 269	187 433
Longitud media			
Pareadas		509	445
Simples		1 195	550
Lecturas rechazadas	#2		
Pareadas		48 403 (32,6 %)	238 804 (55 %)
Simples		49 530 (16,7 %)	53 251 (28,4 %)
Contaminación			
Pareadas		21556 (14,5 %)	62761(14,5 %)
Simples		46766 (15,7 %)	37791 (20,1 %)
Total de lecturas útiles	#3	**382 755**	**396 450**
Lecturas pareadas	#4	**69 318 (23,3 %)**	**132 550 (15,3 %)**
Lecturas simples		**313 437**	**263 900**
Desde la librería de pareadas (Lecturas no emparejadas)		65 553 (44,1 %)	129 264 (29,8 %)
Desde la librería de simples		247 884 (83,4 %)	134 636 (71,8 %)

IV.3.1.2. Ensamblaje

El primer genoma que ensamblamos fue el de L091106-03H, ya que fue el primero del cual recibimos los datos de secuenciación. El conocimiento del tamaño del genoma de DI21 (4,77 Mb) nos permitió hacer una aproximación de la cobertura de las lecturas de L091106-03H que resultó ser baja (14x). Para realizar el proceso de ensamblaje de estas lecturas, y en función de los resultados generados en las pruebas sobre lecturas genómicas de tipo Roche/454 (apartado IV.1.2.2), se seleccionó el programa CABOG [54], ya que es el más preciso en los casos donde las coberturas son bajas. La estrategia de ensamblaje utilizada se ilustra en la figura IV.43, donde CABOG generó 510 contigs y 25 *scaffolds*, los cuales formaron la versión 1 del borrador de genoma de L091106-03H.

IV.3.1.3. Mejora del ensamblaje

Para mejorar el ensamblaje de L091106-03H, los *scaffolds* procedentes de CABOG se trataron con otros dos programas: SSPACE [121] para unificar los *scaffolds* (utiliza un algoritmo de unificación diferente al de CABOG), el cual había sido recomendado en un estudio anterior de comparación de programas de *scaffolding* [188], y SOAPdenovo GapCloser [66] para rellenar los huecos (*gaps*) con nucleótidos. Como la resolución de los huecos con nucleótidos permite que nuevas lecturas pareadas mapeen sobre los *scafffolds*, la información de conexiones entre éstos últimos aumenta, y en consecuencia también aumenta la longitud de los nuevos *scaffolds* formados, por lo tanto este ciclo de resolución de los huecos y unificación de *scaffolds* se realiza varias veces. La nueva estrategia de ensamblaje mejorada ilustrada en la figura IV.44 dio lugar a la versión 2 del borrador de genoma de L091106-03H y fue la misma que se utilizó para ensamblar el genoma de DI21

En la tabla IV.26 se muestra el progreso del número de contigs y *scaffolds,* así como el número de N en los *scaffolds* según las etapas del ensamblaje para las dos cepas. En lo que se refiere a L091106-03H, uno de los 25 *scaffolds* fue eliminado por identificarse como secuencia de contaminación (*Mycoplasma agalactiae*) por lo que finalmente quedaron 24 *scaffolds*. Después de los ciclos de unificación con SSPACE intercalados con el relleno de los huecos con SOAPdenovo GapCloser, éstos se pudieron reducir a 14. El mejor borrador de genoma que formó la versión 2 de esta cepa se obtuvo en el segundo ciclo de unificación ya que en el tercero no se ofreció mejoras. Para DI21, CABOG generó 19 *scaffolds* que en este caso no se pudieron reducir de número ya que SSPACE no consiguió establecer conexiones entre ellos, pero sí que pudimos rellenar huecos con SOAPdenovo GapCloser.

Tabla IV.26: *Progreso del genoma ensamblado según las etapas del ensamblaje*

		Cepas	
		L091106-03H (v2)	DI21
CABOG	*scaffolds*	24	19
	SN	436 933	654 392
SoapDenovo GapCloser	*scaffolds*	24	19
	SN	330 760	562 554
SSPACE (unión 1)	*scaffolds*	15	19
	SN	343 769	562 554
SoapDenovo GapCloser	*scaffolds*	15	19
	SN	340 166	561 264
SSPACE (unión 2)	*scaffolds*	14	19
	SN	341 170	561 264
SoapDenovo GapCloser	*scaffolds*	14	19
	SN	341 126	561 264
SSPACE (unión 3)	*scaffolds*	14	-
	SN	341 126	-
SoapDenovo GapCloser	*scaffolds*	14	-
	SN	341 126	-

SN: suma de las N restantes

Características del borrador de genoma de las dos cepas

En tabla IV.27 se muestran las diferentes características del borrador de genoma de las dos cepas de *P. damselae* subsp. *piscicida*. En lo que se refiere a L091106-03H (v2), el borrador de genoma está formado por 14 *scaffolds* con un margen de longitudes de 1 007 a 2 323 982 pb, una longitud media de 299 600 pb y un porcentaje G+C del 40%. Respecto a DI21, el borrador de genoma lo forman 19 *scaffolds* con un margen de longitudes de 435 a 2 798 534 pb, con una longitud media de 227 180 pb y un porcentaje G+C de 40,6%. En ambos borradores, el *scaffold* más largo supera la mitad del genoma (>2 Mb), por lo que el N50 iguala la longitud de este scaffold. Por lo tanto, se puede deducir que el ensamblaje fue equivalente para ambas cepas. Como el borrador del genoma de DI21 en el NCBI (GCA_000300355.3) contenía 56 *scaffolds* con 846 993 indeterminaciones (N), podemos sugerir que, al tener el nuevo borrador solo 19 *scaffolds* y menos N (561 264), se ha mejorado el ensamblaje de esta cepa.

Tabla IV.27: *Características del borrador de genoma final de L091106-03H y DI21*

	Cepas	
	L091106-03H (v2)	**DI21**
Número de *scaffolds* > 500 pb	14	17
El *scaffolds* más largo	2 323 982	2 798 534
El *scaffolds* más corto	1 007	437
Suma de longitudes	4 194 408	4 316 437
Número de N	341 126	561 264
Longitud medía	299 600	227 180
N50	2 323 982	2 798 534
N90	157 598	152 634
Contenido G+C	40%	40,6%

IV.3.1.4. Anotación de los dos borradores de genomas

La anotación de los borradores de genomas de L091106-03H y DI21 de se llevó a cabo con el sistema de anotación automática RAST (Rapid Annotation using Subsystem Technology) [125] (`http://rast.nmpdr.org`), que destacaba en un estudio reciente [189] de comparación de programas de anotación para genomas bacterianos. En L091106-03H (v2), el número de secuencias codificantes fue de 3 625 y se identificaron 122 ARNs, mientras que en DI21 el número de secuencias codificantes fue 3 527 y se identificaron 97 ARNs. En la figura IV.45 se muestran los genes conectados a los diferentes subsistemas y su distribución en las diferentes categorías. En total, el 54% de los genes de L091106-03H se pudieron clasificar en 458 subsistemas (figura IV.45-A), y el 55% de los genes de DI21 se pudieron clasificar en 456 subsistemas (figura IV.45-B).

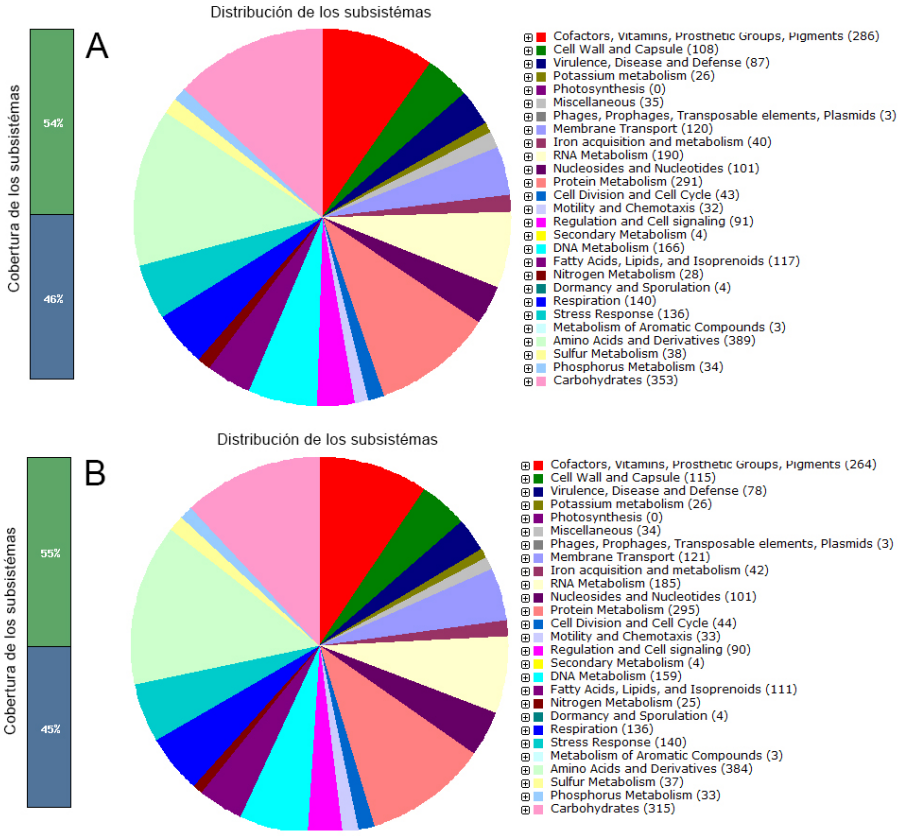

Figura IV.45: *Distribución de los subsistemas del genoma L091106-03H v2 (A) y DI21 (B) sobre la base del sistema de anotación automática RAST [125]. La barra verde de la cobertura de los subsistemas indica el porcentaje de proteínas incluidas en los subsistemas mientras que la barra azul se refiere al porcentaje de proteínas que no fueron incluidas en los subsistemas*

Desde la web de RAST, los datos de anotación se exportaron a un fichero GFF (`https://www.sanger.ac.uk/resources/software/gff/`). La base de datos PhotobacteriumDB (`http://www.scbi.uma.es/photobacterium_damselae/`) fue desarrollada en nuestro laboratorio sobre la base de GBROWSE [190] para importar los datos de anotación del genoma del genoma de *Photobacterium damselae* incluidos en el fichero GFF y visualizarlos de forma interactiva. En la figura IV.46 se muestran algunas capturas de pantalla de esta web que por ahora solo incluye los datos de anotación de L091106-03H (v1).

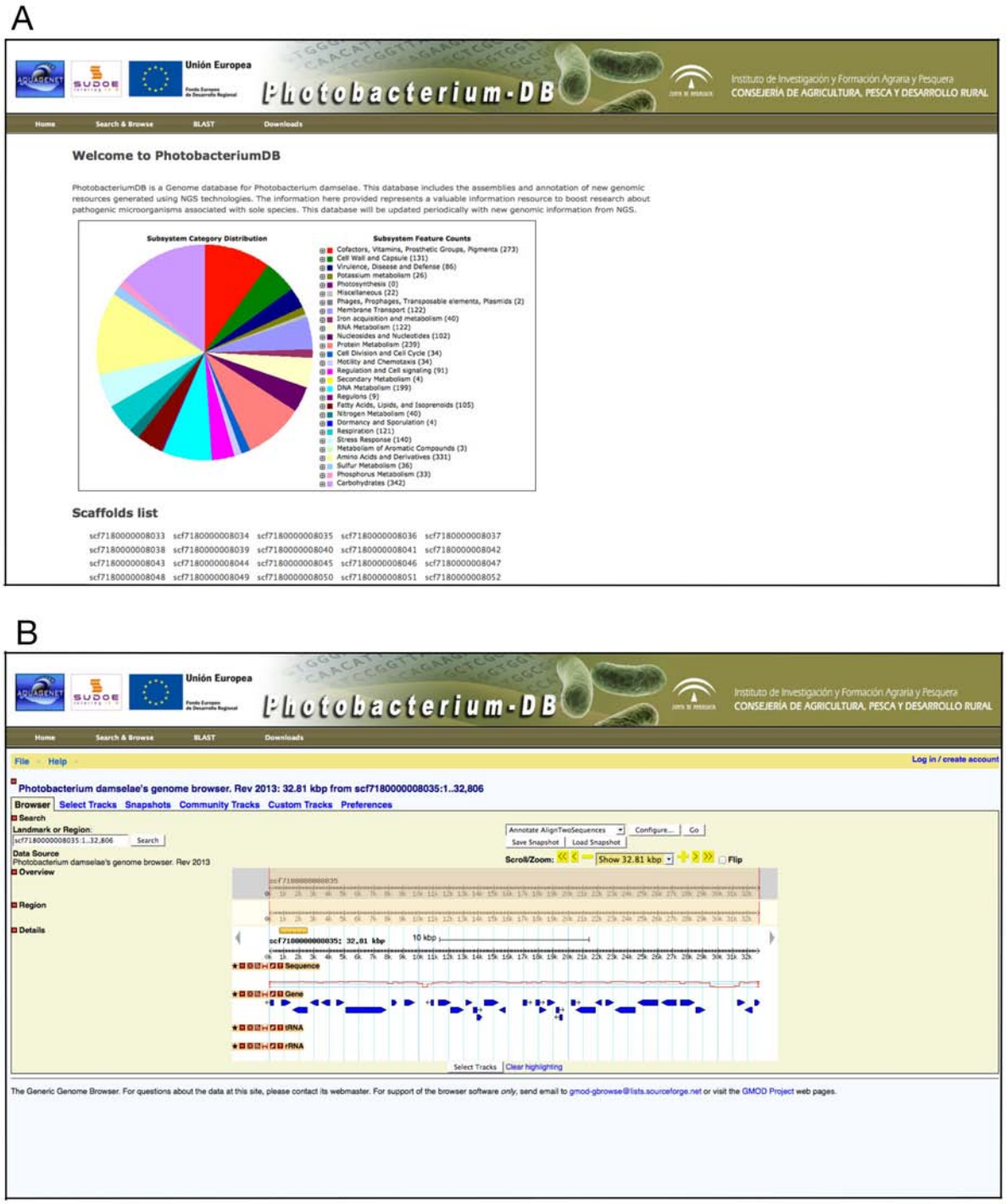

Figura IV.46: *Capturas de pantalla de la interfaz de la base de datos PhotobacteriumDB. A: Portada de la web. B: Herramientas interactivas de GBROWSE [190] que permiten buscar y visualizar las diferentes regiones del genoma y los genes localizados en estas regiones*

IV.3.1.5. Análisis comparativo entre las dos cepas de *P. damselae* subsp. *piscicida*

Una comparación entre las diferentes categorías de los subsistemas en los genomas de L091106-03H v2 (figura IV.45-A) y DI21 (figura IV.45-B) muestra que el número de genes contenidos en las diferentes categorías son bastante equivalentes entre las dos cepas. Las proteínas de L091106-03H v2 se compararon con las de otras bacterias, incluida la cepa DI21. En el diagrama de la figura IV.47 se muestran los porcentajes de identidad de los alineamientos con las proteínas de cada bacteria. Se observa que los porcentajes de identidad fueron bajos para *E. coli* pero altos (identidad de casi el 100%) para las especies *Photobacterium damselae* especialmente para la cepa DI21, lo que confirmaba que los dos genomas de L091106-03H v2 y DI21 parecen ser muy similares.

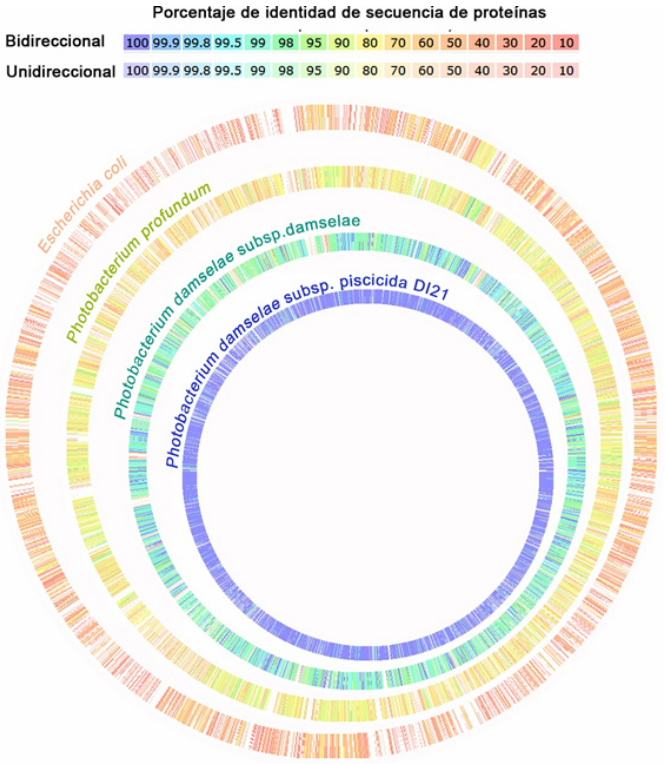

***Figura IV.47**: Similitud entre L091106-03H v2 y otras bacterias basada sobre el porcentaje de identidad del alineamiento entre las proteínas*

Para evaluar el grado de similitud entre los genomas de L091106-03H v2 y DI21 se utilizaron dos estrategias: la primera se basó en la comparación de las secuencias de ADN y consistió en alinear los dos borradores de genoma con Blastn. En la figura IV.48 se muestra el dotplot de este alineamiento en el cual se nota la existencia de repeticiones entre los dos genomas que seguramente se refieren a transposones. Igualmente, se nota que el *scaffold* 1 de L091106-03H v2 y el *scaffold* 1 de DI21 que son los *scaffolds* más largos y se encuentran bien alineados entre sí. El *scaffold* 1 de DI21 se alineó también con los *scaffold* 3 y 9 de

L091106-03H v2. De igual forma, se formaron alineamientos entre los demás *scaffolds* de las dos cepas, lo que indica la existencia de una similitud nucleotídica entre ambas cepas que cubre sus genomas completos, pese a que haya algunas pequeñas inversiones, como la inversión 1 en el alineamiento entre el *scaffold* 1 de L091106-03H v2 y el *scaffold* 1 de DI21 o la inversión 2 en el alineamiento entre el *scaffold* 5 de L091106-03H v2 y el *scaffold* 7 de DI21 (figura IV.48), las cuales pueden explicarse tanto por una reorganización ocurrida en los genomas o por posibles errores de ensamblaje en uno de ellos. La segunda estrategia consistió en correlacionar los *scaffolds* de L091106-03H v2 con los de DI21 a través del posicionamiento de las proteínas de *Photobacterium damselae* (descargadas desde NCBI) sobre los *scaffolds* de ambas cepas. En la figura IV.49 se ilustra el protocolo utilizado para establecer esta correlación. El resultado ilustrado en el diagrama de CIRCOS [191] de la figura IV.50 muestra que los genes de DI21, tanto aquellos contenidos en el *scaffold* más largo como aquellos contenidos en los demás *scaffolds*, se encuentran en el mismo orden en el genoma de L091106-03H v2. Sin embargo, en el diagrama se aprecia que algunos *scaffolds* de DI21 tienen conexiones con varias zonas del genoma de L091106-03H, lo cual muestra que algunas proteínas de *Photobacterium damselae* se encuentran repetidas en el genoma de L091106-03H. De cualquier modo, en un alineamiento tblastn entre las proteínas de *Photobacterium damselae* y los dos genomas, comprobamos que existen proteínas que alinean en varios sitios también del genoma de DI21 (aunque no es tan evidente en el diagrama). A titulo de ejemplo, las dos proteínas hipotéticas WP_005304266 y WP_005304755 se alinean (completamente y con más de 98% de porcentaje de identidad) en 4 posiciones distintas tanto en L091106-03H como en DI21.

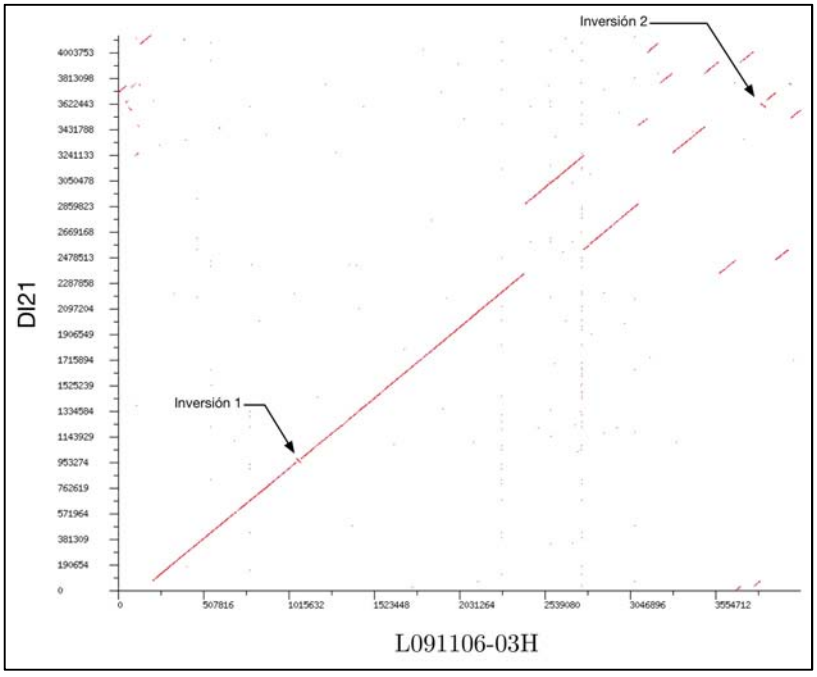

Figura IV.48: *Representación dotplot de los alineamientos nucleotídicos entre L091106-03H (v2) y DI21*

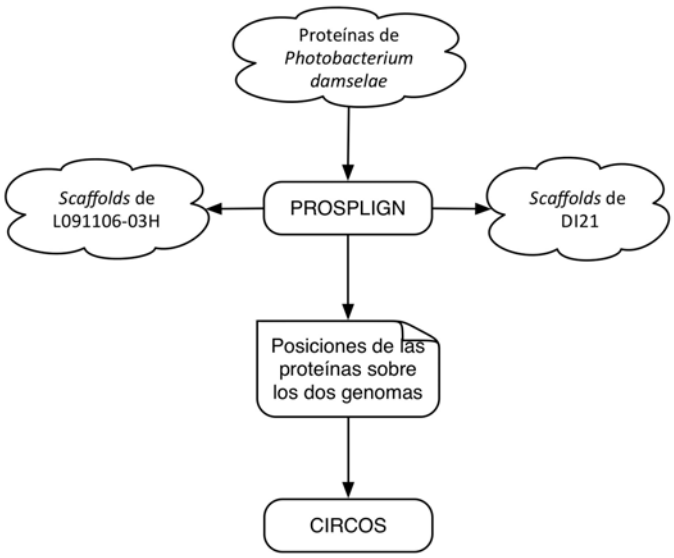

Figura IV.*49*: *Método utilizado para correlacionar los scaffolds de L091106-03H v2 con aquellos de DI21 basado sobre el posicionamiento de las proteínas de Photobacterium damselae (descargadas desde NCBI) sobre ambas cepas con el programa PROSPLIGN (*`http://www.ncbi.nlm.nih.gov/sutils/static/prosplign/prosplign.html`*). CIRCOS hace referencia al programa que se usó para visualizar los resultados (véase figura siguiente)*

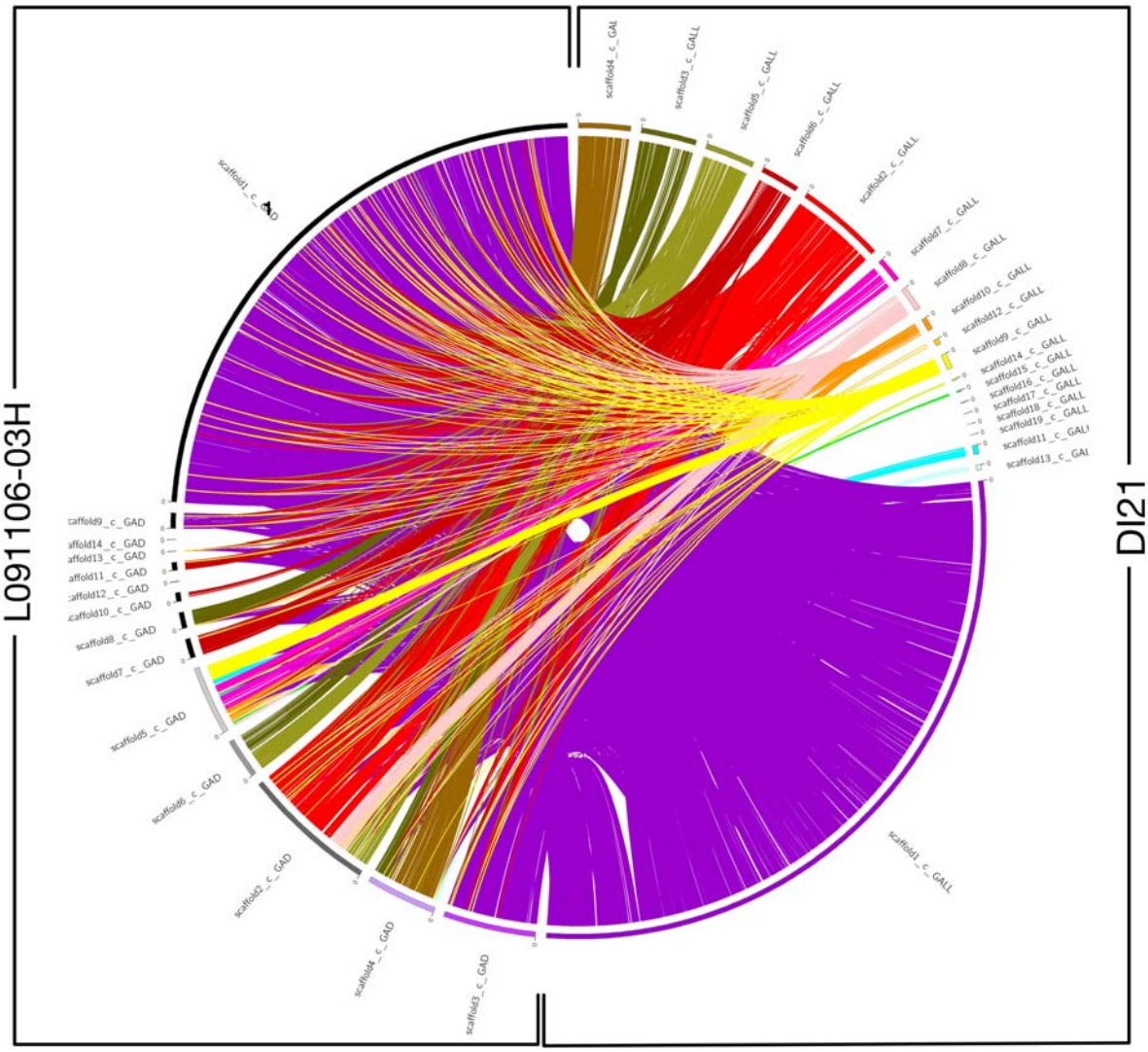

Figura IV.50: *Visualización de la sintenia entre los borradores de genomas de L091106-03H v2 y DI21 en función de la correspondencia obtenida con las proteínas de Photobacterium damselae (identidad mínima del 97% en ambas especies). Las coincidencias que se refieren a las posiciónes de una misma proteína en DI21 y en L091106-03H fueron representadas con Circos [191]. Los genes de DI21, tanto aquellos contenidos en el scaffold más largo (conexiones de color morado) como aquellos contenidos en los demás scaffolds, se encuentran en el mismo orden en el genoma de L091106-03H v2.*

Para comprobar la colinealidad entre los genes de las dos cepas, la disposición de los CDS en los genomas de L091106-03H v2 y DI21 fue observada utilizando SEED Viewer [192], que está integrado con el programa de anotación RAST (`http://rast.nmpdr.org`). En la figura IV.51 se muestran dos ejemplos de la disposición de dos grupos de genes ortólogos en los genomas de las cepas. En el primer ejemplo (figura IV.51-A) se observa que el orden de los genes ortólogos está bien conservado entre las dos cepas mientras que en el segundo ejemplo (figura IV.51-B) se nota que el grupo de genes ortólogos (6, 22, 31, 30, 29, 32 y 35) está localizado en medio de otros genes que son diferentes lo que indica que este grupo de

genes ortólogos se encuentra en dos zonas distintas entre los *scaffolds* 5 de L091106-03H v2 y el *scaffold* 11 de DI21, además se nota que el orden de estos genes ortólogos no está conservado ya que el gen 30 tiene una posición relativa diferente entre los dos genomas. La ocurrencia de esta figura fue muy rara pero confirma la hipótesis de que hubieron algunas reorganizaciones en los genomas durante la evolución de las dos cepas. En cambio, la primera figura, donde el orden de los genes ortólogos esta conservado, fue la más predominante indicando que en general los genomas de las dos cepas son colineales.

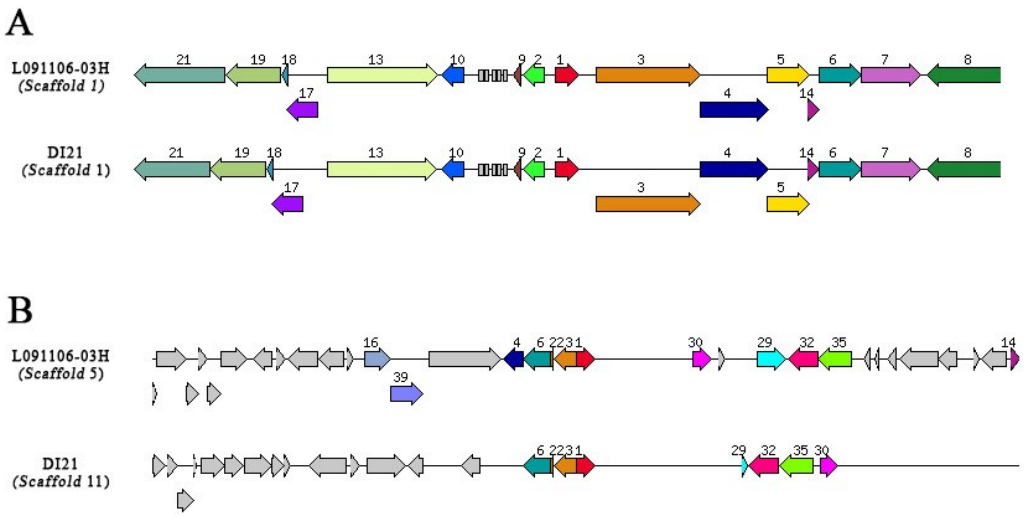

Figura IV.51: *Ejemplos de localización de los genes ortólogos entre L091106-03H v2 y DI21. Las regiones mostradas tienen una longitud de 40kb. La correspondencia entre las regiones de los dos genomas se basó sobre la posición del gen numerado 1 en el genoma, luego los genes vecinos fueron situados conservando su posición relativa. Los genes que no tienen ortólogo en la zona mostrada están marcados con color gris y los genes ortólogos están marcados con color similar que es diferente del gris. Los solapamientos se resuelven dibujando la flecha solapante en una nueva línea. En el primero ejemplo (A) los genes ortólogos tienen el mismo orden y la misma posición relativa entre las dos cepas, mientras que en el segundo ejemplo (B) el grupo genes que son ortólogos están situados en medio de otros genes que no son correspondientes*

IV.3.1.6. Descubrimiento de los plásmidos de L091106-03H

Los plásmidos bacterianos son elementos extracromosómicos autorreplicantes asociados a muchas características fenotípicas que incluyen la virulencia y la resistencia a los agentes antimicrobianos.

Como el genoma de L091106-03H se parece mucho al de DI21, era probable que L091106-03H también tenga los plásmidos de DI21. El ensamblaje de las lecturas de L091106-03H parecidas a los plásmidos de DI21 resultó muy fragmentado (seguramente porque la cobertura de las lecturas era baja o poco homogénea), por lo que no fue posible aislar las secuencias de los plásmidos. Para saber cuáles de los plásmidos de DI21 (pPHDP10, pPHDP60 y pPHDP70) existen también en L091106-03, realizamos un mapeo con BOWTIE

de todas las lecturas de L091106-03H sobre las secuencias de los 3 plásmidos de forma independiente. La cobertura de las lecturas sobre los plásmidos se ilustra en la figura IV.52. Se observa que la secuencia del plásmido pPHDP10 fue completamente cubierta con las lecturas de L091106-03H (figura IV.52-A), lo que confirma que este plásmido existe en L091106-03H. Sin embargo, los otros 2 plásmidos (pPHDP60; figura IV.52-B y pPHDP70; figura IV.52-C) solo presentaban algunos picos de cobertura que seguramente corresponden a una superposición de lecturas de transposones, pero la mayor parte de sus secuencias carecía de lecturas mapeadas. Habría dos explicaciones de esto: la primera es que L091106-03H no tiene estos 2 plásmidos y la segunda que L091106–03H tiene estos 2 plásmidos, pero no estaban en la muestra que se secuenció. En el laboratorio de Juan José Borrego (Dpto de Microbiología, UMA) comprobaron que la cepa L091106–03H sí tiene los otros plásmidos, y se han enviado a secuenciar recientemente.

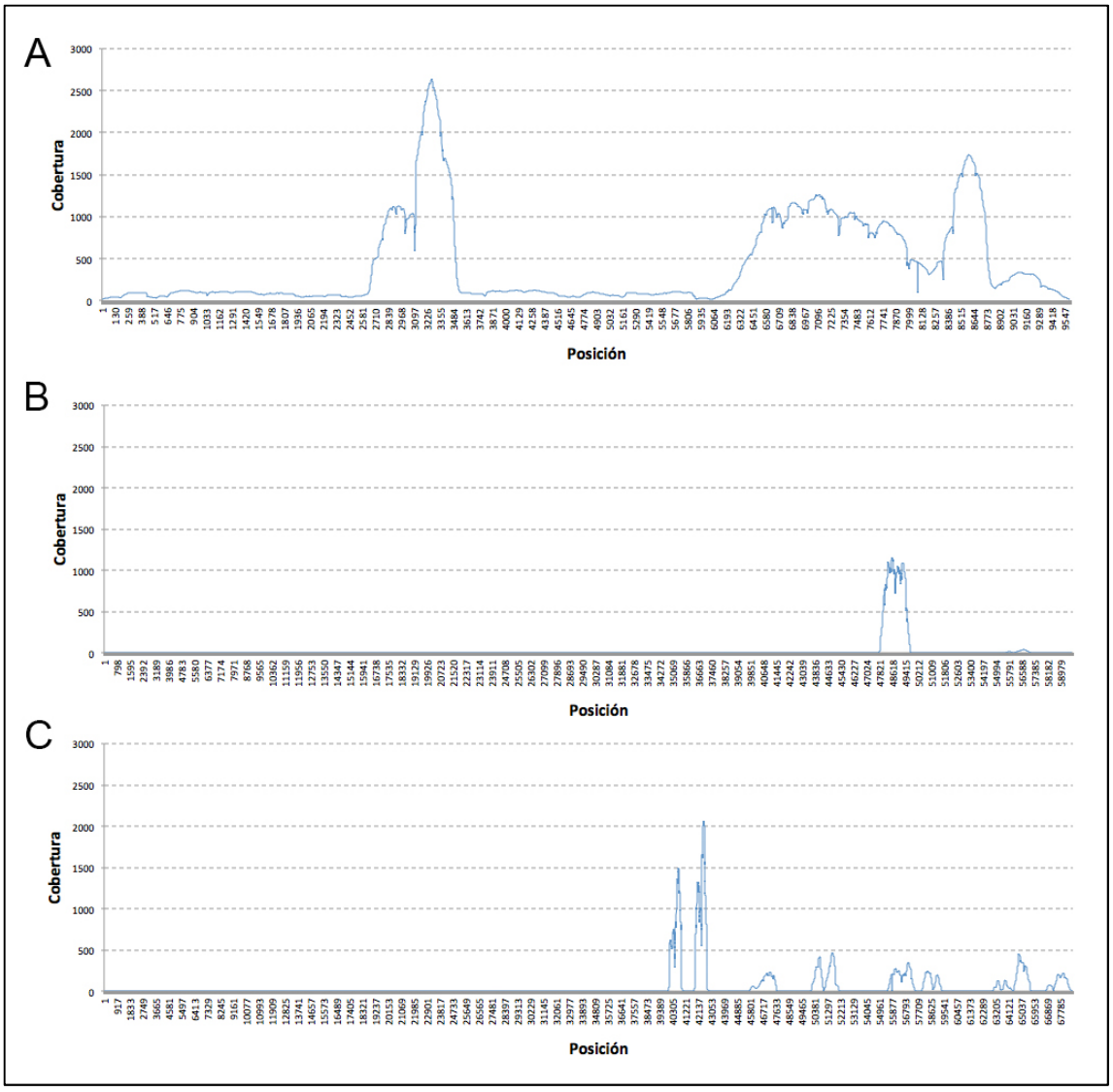

Figura IV.52: *Cobertura de las lecturas de L091106-03H sobre las secuencias de los plásmidos de DI21. A: pPHDP10, B: pPHDP60 y C: pPHDP70*

IV.3.2. Genoma del lenguado senegalés (*Solea senegalensis*).

La secuenciación y ensamblaje del transcriptoma de *S. senegalensis* descrita en este trabajo ha ofrecido a la comunidad científica una información valiosa sobre los genes de esta especie y sus funciones. En este apartado, vamos a ir un paso más hacia delante, continuando con el estudio bioinformático de la genómica de *S. senegalensis* y presentado el ensamblaje del genoma completo de este pez. La secuenciación del genoma completo es la clave para entender algunos mecanismos biológicos importantes como son: la determinación del sexo, la metamorfosis, y los mecanismos evolutivos; por otro lado, permitirá a los genéticos tener un mapa con la asignación espacial de las variaciones genéticas (SNP) y así poder realizar selecciones asistidas con el fin de aumentar la resistencia a enfermedades y la producción.

IV.3.2.1. Preprocesamiento de las lecturas genómicas brutas

Se prepararon 45 librerías Illumina y 14 librerias de Roche/454 en el laboratorio del grupo de investigación de IFAPA El Toruño (Cádiz), todas desde de muestras de hembras. Los resultados de preprocesamiento con SeqTrimNext (figuras IV.53, IV.54 y IV.55) de las lecturas Roche/454 (largas y pareadas) e Illumina (cortas y pareadas) se muestran en la tabla IV.28. Se observa que en todas las librerías los porcentajes de contaminación fueron muy bajos. Sin embargo en las lecturas Roche/454 se nota un porcentaje grande de lecturas rechazadas causadas por las repeticiones producidas por la máquina de secuenciación Roche/454 (fenómeno conocido como clonalidad de lecturas), las cuales fueron más notables en las librerías de 8 kb y de 20 kb (39,4% y 64,2% respectivamente). El porcentaje de lecturas pareadas fue bastante bajo en las librerías 3 kb (48%) y muy bajo en las librerías de 8 kb y 20 kb (12,5% y 0,9% respectivamente). Esta baja tasa de lecturas pareadas procede de errores en el proceso circularización del ADN debido a la abundancia de los nucleótidos GC en el genoma de *S. senegalensis*, según nos indicaron los técnicos de Life Sequencing (F. Codoñer, comunicación personal). Respecto a las lecturas de Illumina, en todas las librerías se recuperó un porcentaje alto de lecturas limpias, la gran mayoría de ellas fueron pareadas.

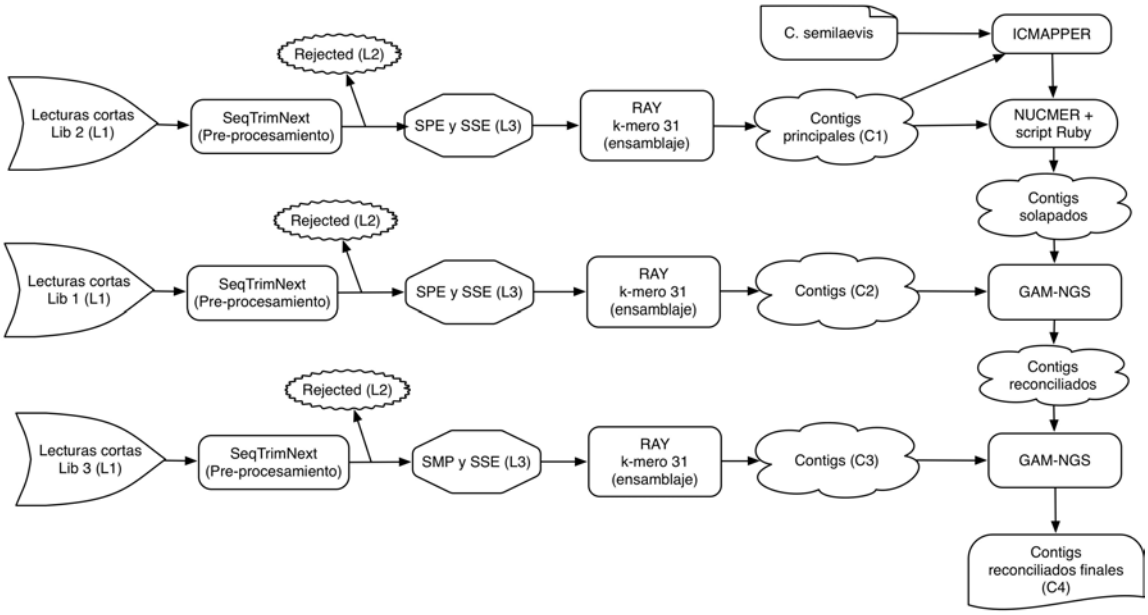

Figura IV.53: *Preprocesamiento, ensamblaje y reconciliación de los contigs. SPE: lecturas cortas tipo paired-end, SMP: lecturas cortas tipo mate-pair, SSE: lecturas cortas simples*

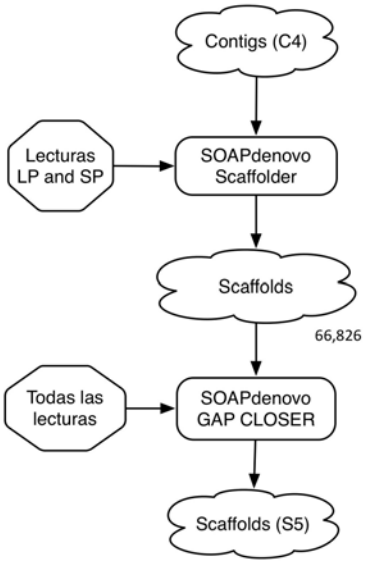

Figura IV.54: *Primera etapa de scaffolding*
LP: lecturas largas pareadas. SP: lecturas cortas pareadas

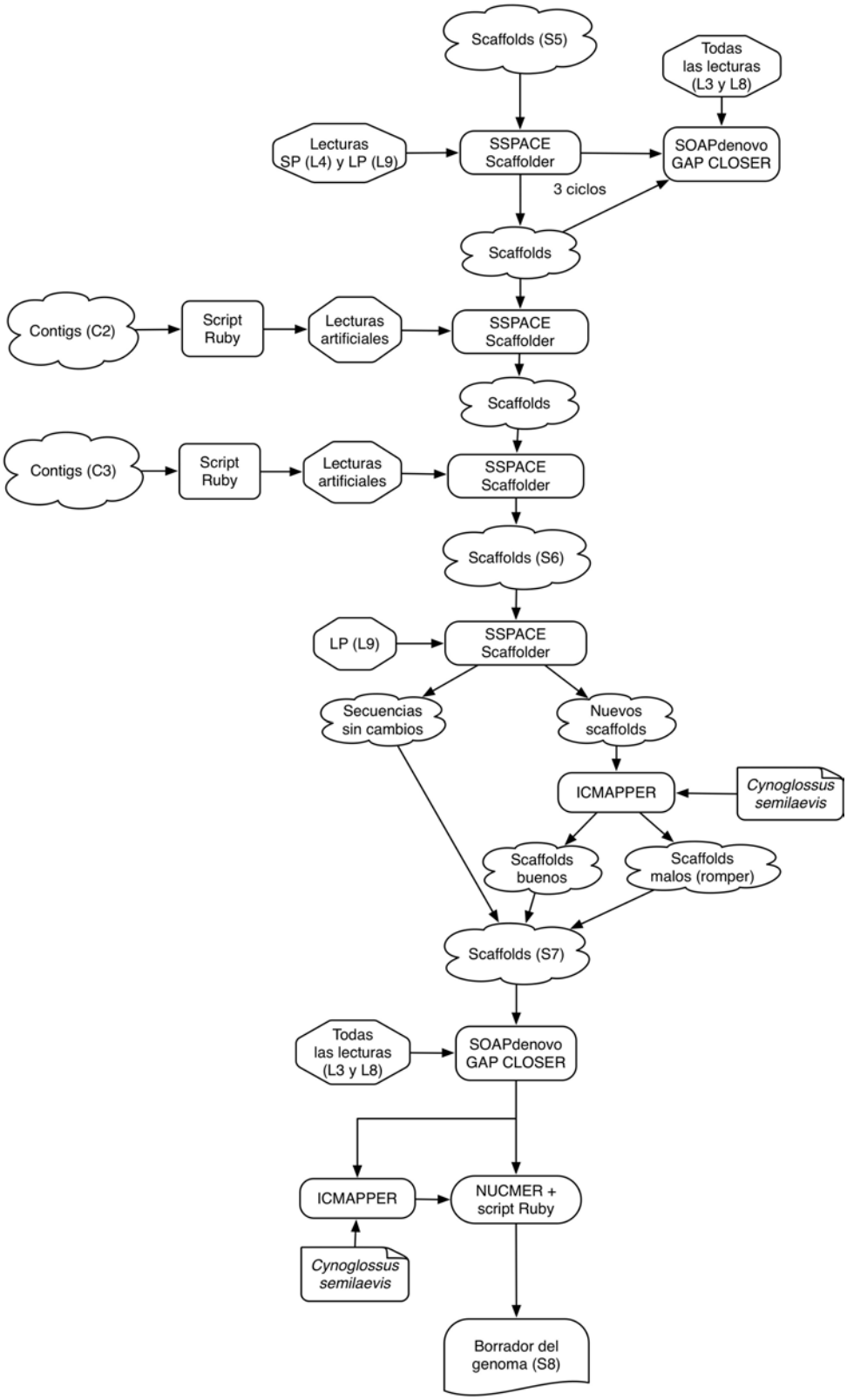

Figura IV.55: *unificación de los contigs con SSPACE. Se utilizaron diferentes estrategias de unificación con SSPACE con diferentes tipos de lecturas. Posteriormente se realizó un relleno de N con* SOAPdenovo GapCloser *y una reconciliación con NUCMER.*
LP: Lecturas pareadas largas. SP: Lecturas pareadas cortas

Tabla IV.28: *Resumen del preprocesamiento de las lecturas originales del genoma del lenguado senegalés*

			Illumina		
	Referencia a las figuras IV.53, IV.54 y IV.55		**Lib 1**	**Lib 2**	**Lib 3**
Tamaño de inserto			300	300	3kb
Lecturas brutas totales	L1		869 031 542	1 005 527 592	1 109 021 114
Longitud media bruta			150	100	101
Rechazos (total)	L2	N	44 137 604	21 257 133	59 706 278
		%	5,1	2,1	5,4
Contaminación		N	3 345 627	1 336 437	2 161 966
		%	0,4	0,1	0,2
Total de lecturas útiles	L3	N	824 893 938	984 270 459	1 049 314 836
		%	94,9	97,9	94,6
Lecturas pareadas	L4	N	795 887 280	966 158 260	1 022 901 978
		%	91,6	96,1	92,2
Lecturas simples	L5	N	29 006 658	18 112 199	26 412 858
		%	3,3	1,8	2,4
Longitud media limpia			106	91	86

			454		
	Referencia a las figuras IV.53, IV.54 y IV.55		**Lib 1**	**Lib 2**	**Lib 3**
Tamaño de inserto			3kb	8kb	20kb
Lecturas brutas totales	L6		2 103 984	5 236 315	1 018 187
Longitud media bruta			551	546	579
Rechazos (total)	L7	N	254 981	2 244 068	675 260
		%	12,1	42,9	66,3
Contaminación		N	3 476	22 887	1 549
		%	0,2	0,4	0,2
Repetidas		N	221 372	2 065 550	653 190
		%	10,5	39,4	64,2
Total de lecturas útiles	L8	N	2 859 068	3 647 614	351 729
		%	87,9(*)	57,1(*)	33,7(*)
Lecturas pareadas	L9	N	2 020 130	1 310 734	17 604
		%	48,0(*)	12,5(*)	0,9(*)
Lecturas simples	L10	N	838 938	2 336 880	334 125
		%	39,9	44,6	32,8
Longitud media limpia			167	186	222

(*) En el cálculo de este porcentaje se cuentan los pares de lecturas pareadas.

IV.3.2.2. Ensamblaje

Debido al gran número de lecturas de Illumina, se probaron varios programas de ensamblaje descritos con uso eficiente de la memoria, utilizando como prueba la librería 1 (tabla IV.29). Los resultados revelaron que el programa RAY proporcionaba el mejor compromiso entre capacidad de ejecución y resultados, aunque no era capaz de generar *scaffolds* para el k-mero 31 debido a que sobrepasaba constantemente el limite de tiempo. Los otros ensambladores produjeron un número alto de contigs y *scaffolds,* lo que constituía un genoma bastante fragmentado.

Tabla IV.29: *Pruebas de ensamblaje usando 828 254 151 lecturas útiles de la Librería 1 (AQG1) con dos diferentes k-meros*

Ensamblador	k-mero 31		k-mero 51	
	Contigs	Scaffolds	Contigs	Scaffolds
RAY	263 123	LT	582 421	562 157
Minia	910 625	ND	430 128	ND
SOAPdenovo	21 469 279	1 429 112	10 716 892	2 058 361
Platanus	ND	ND	2 822 377	1 259 857

LT: Valor no calculado debido a la excepción del limite de tiempo. ND: No disponible

El ensamblaje del genoma se dividió en dos etapas principales: en la primera se ensamblaron las lecturas en contigs con RAY [68] (figura IV.53) y en la segunda se unificaron los contigs en *scaffolds* con SOAPdenovo Scaffolder [66] (figura IV.54) y SSPACE [121] (figura IV.55). Adicionalmente se utilizaron los programas NUCMER [134] y GAM-NGS [118] para la reconciliación de los contigs y *scaffolds* (figura IV.53), SOAPdenovo GapCloser [66] para rellenar las N con nucleótidos (figuras IV.54 y IV.55), e ICMapper (apartado IV.1.4) para ordenar los contigs o *scaffolds* al fin de validar su posterior unificación. En la tabla IV.30 se muestran el número de contigs y *scaffolds* obtenidos después de cada etapa del proceso de ensamblaje.

Tabla IV.30: *Número de contigs y scaffolds obtenidos después de cada etapa del proceso de ensamblaje*

Etapa	**Contigs (Referencia a las figuras IV.53, IV.54 y IV.55)**	**Scaffolds (Referencia a las figuras IV.54 y IV.55)**
Ensamblaje con RAY		
Librería 2*	213 548 (C1)	
Librería 1	263 123 (C2)	
Librería 3	278 995 (C3)	
Reconciliación con NUCMER + GAM-NGS	132 710 (C4)	
Scaffolding con lecturas cortas utilizando SOAPdenovo Scaffolder		66 826 (S5)
Scaffolding con lecturas cortas utilizando SSPACE		36 524 (S6)
Scaffolding con lecturas largas utilizando SSPACE		34,857 (S7)
Reconciliación final con NUCMER		34,176 (S8)

(*) Los contigs de la librería 2 se utilizaron como contigs de referencia debido a que la librería 2 contenía el mejor porcentaje de lecturas útiles y que sus contigs no tenían secuencias contaminantes.

Generación de los contigs

En la figura IV.53 se muestra la estrategia utilizada en la generación de contigs. Cada librería fue ensamblada con RAY con un k-mero = 31 para obtener su correspondiente colección de contigs (tabla IV.30, filas "Ensamblaje con RAY").

Para extender los contigs de la librería 2 (tomados como contigs principales) y eliminar las redundancias, se utilizó un script que desarrollamos en nuestro laboratorio, que parte del resultado de alineamiento con NUCMER para unir los contigs solapantes. En el solapamiento se utilizaron criterios estrictos (100% de identidad y solapamiento >300 nt o 99% de identidad y solapamiento >2 000 nt). Cuando se utilizaron otros criterios de solapamiento menos estrictos, fue necesario confirmar la contigüidad de los contigs con ICMapper aprovechándonos de la enorme sintenia entre el lenguado senegalés y el lenguado *Cynoglossus semilaevis*.

La combinación de los contigs procedentes de las tres librerías de lecturas se realizó mediante GAM-NGS que proporcionó un total 132 710 contigs (tabla IV.30)

Unificación de los contigs en scaffolds

Para unir los contigs en *scaffolds*, se utilizaron SOAPdenovo Scaffolder (figura IV.54) y SSPACE (figura IV.55). Las etapas de *scaffolding* fueron intercaladas con un relleno de N con SOAPdenovo GapCloser. A la diferencia de SOAPdenovo Scaffolder que solo permite unir contigs, SSPACE también permite unir *scaffolds*, lo que hizo posible utilizarlo en varías estrategias consecutivas de unificación de los *scaffolds* (figura IV.55).

Los *scaffolds* procedentes de SOAPdenovo Scaffolder se pasaron a SSPACE para una segunda etapa de unificación donde se utilizaron las lecturas pareadas en 3 ciclos consecutivos, intercalados con un relleno de N con SOAPdenovo GapCloser. El relleno de las N con nucleótidos permite que mapeen nuevas lecturas pareadas y con ello se formen más conexiones entre los *scaffolds*.

SSPACE se utilizó en una etapa posterior, donde se sirvió de lecturas artificiales construidas a partir de las librerías 1 y 3 con longitudes de 800 nt e insertos de 3 000, 8 000, 20 000, 30 000, 40 000 y 50 000 nt, para unir más *scaffolds*.

En una última etapa de unificación de los *scaffolds* con SSPACE (figura IV.55), se utilizaron las lecturas largas (Roche/454) donde se consideró que el número de pareadas mínimas mapeadas bastaba que fuera 1 (el valor por defecto era 5) por dos motivos: porque teníamos pocas pareadas y comprobamos que ni tan siquiera había 3 pareadas para conectar *scaffolds*, y porque al ser más largas, la coincidencia podía ser más fiable que con las de Illumina. Como esta unificación fue poco estricta (podría ser que la pareada mapeara incorrectamente, por ejemplo, sobre regiones repetitivas), los nuevos *scaffolds* formados de esta manera se validaron con ICMapper utilizando la sintenia con *Cynoglossus semilaevis*. Los *scaffolds* que no se eran consecutivos en ambas especies se volvieron a separar.

Después de un relleno de las N con SOAPdenovo GapCloser, se realizó una reconciliación final con NUCMER combinado con el script Ruby (verse el apartado III.3.5.7 de material y métodos) donde se utilizaron criterios más estrictos que antes (100% de identidad y un solapamiento >500 nt o 99% de identidad y un solapamiento >2 000 nt) o criterios menos estrictos, validando la contigüidad de los *scaffolds* con su orden proporcionado por ICMapper al igual que antes (figura IV.55).

El borrador de genoma final incluyó a 34 176 *scaffolds* con una suma total de 600 313 426 pb y una longitud media de *scaffolds* de 17 565 y un N50 de 85 596 nt (tabla IV.31). Hay que señalar que el tamaño de genoma ensamblado estaba muy cerca al tamaño de genoma predicho por el programa KmerGenie [64] para el genoma de lenguado senegalés basándose en la información de las lecturas, que fue de 612 299 319 pb y está bastante cerca del tamaño teórico (713 megabases) (`http://www.genomesize.com`). En comparación con los genomas de otros peces, el genoma ensamblado de *S. senegalensis* es más pequeño que el genoma de la tilapia (927 megabases) y mucho más pequeño que el del pez cebra (1 371 megabases), pero es bastante más grande que el genoma de *Cynoglossus semilaevis* (470 megabases).

Tabla IV.31: *Estadísticas relevantes del borrador de genoma del lenguado senegalés. Los datos se presentan para la colección global de scaffolds representada en la tabla IV.30 (columna "Scaffolds"), y para subgrupos de los scaffolds según su posicionamiento sobre el genoma del lenguado C. semilaevis*

	Todos	Posicionados sobre los cromosomas de *C. semilaevis*	Redundantes o no posicionados
Número de *scaffolds*	34 176	7 971	26 205
Número de *scaffolds* > 500 pb	26 368	7 899	18 469
El *scaffolds* más largo	638 263	58 273	343 325
Suma de longitudes	600 313 426	464 497 837	135 815 589
Número de N	14 313 038	9 777 016	4 536 022
Longitud media	17 565	58 273	5 182
N50	85 596	109 513	24 414
N90	13 932	30 884	2 234

IV.3.2.3. Evaluación del genoma ensamblado

IV.3.2.3.1. Prueba de compleción de los genes en el ensamblaje

La fiabilidad del genoma ensamblado se estimó con la estrategia de mapeo de genes eucarióticos conservados (CEG) con CEGMA [193]. CEGMA se considera como un programa de referencia para la validación de los genomas de organismos no modelo ensamblados *de novo*. Utiliza un conjunto de 248 de genes eucarióticos altamente conservados presentes en una amplia gama de eucariotas (desde la levadura hasta el humano) y que se presentan en una sola copia en todos los genomas de eucariotas. De los 248 CEG buscados en el genoma ensamblado del lenguado senegalés, se encontraron 209 (84,27%) completos y 242 (97,6%) parciales o completos (tabla IV.32). Estos porcentajes de presencia y completitud se pueden considerar altos, con lo que se confirma la fiabilidad del genoma ensamblado.

Tabla IV.32: *Resumen del informe de CEGMA sobre los CEG encontrados en el genoma ensamblado*

	Proteínas	%	Total	Promedio	Ortólogos (%)
Completos	209	84,3	274	1,3	23
Parciales/completos	242	97,6	385	1,59	38,8

IV.3.2.3.2. Confirmación de la sintenia entre los lenguados

En el IFAPA El Toruño han visto que los *scaffolds* genómicos del lenguado senegalés que contienen apolipoproteínas tenían una sintenia importante con los teleósteos (Román-Padilla et al, en preparación), pero especialmente con el lenguado *Cynoglossus semilaevis* y el pez espinoso (*Gasterosteus aculeatus*). La sintenia era menor con el pez modelo (*Danio rerio*). Por lo tanto, se estudió más a fondo la correlación entre los genomas del lenguado senegalés y de *C. semilaevis* con los 9 *scaffolds* más largos del borrador de genoma del lenguado senegalés y la secuencia de los cromosomas de *Cynoglossus semilaevis* en GCF_000523035.1. En la figura IV.56 se muestra el método utilizado para establecer la correspondencia entre los *scaffolds* genómicos de uno con los cromosomas del otro. Un método se basó en las proteínas de *Cynoglossus semilaevis* (figura IV.56-A) y otro en los transcritos del lenguado senegalés obtenidos en el apartado IV.2.1 de esta tesis (figura IV.56-B). Ambas estrategias indican que la mayoría de los genes contenidos en los *scaffolds* del lenguado senegalés se encuentran juntos y en el mismo orden en los cromosomas (figura IV.57), donde se ve claro que las correspondencias entre el *scaffold* 755 y el cromosoma 10, así como entre el *scaffold* 678 y el cromosoma 13 y entre el *scaffold* 695 y el cromosoma 8 que se observan con las proteínas (figura IV.57–A) fueron espurias, y que todos los genes de los *scaffolds* estaban en la misma disposición sobre el genoma de *Cynoglossus semilaevis* (figura 3.57-B).

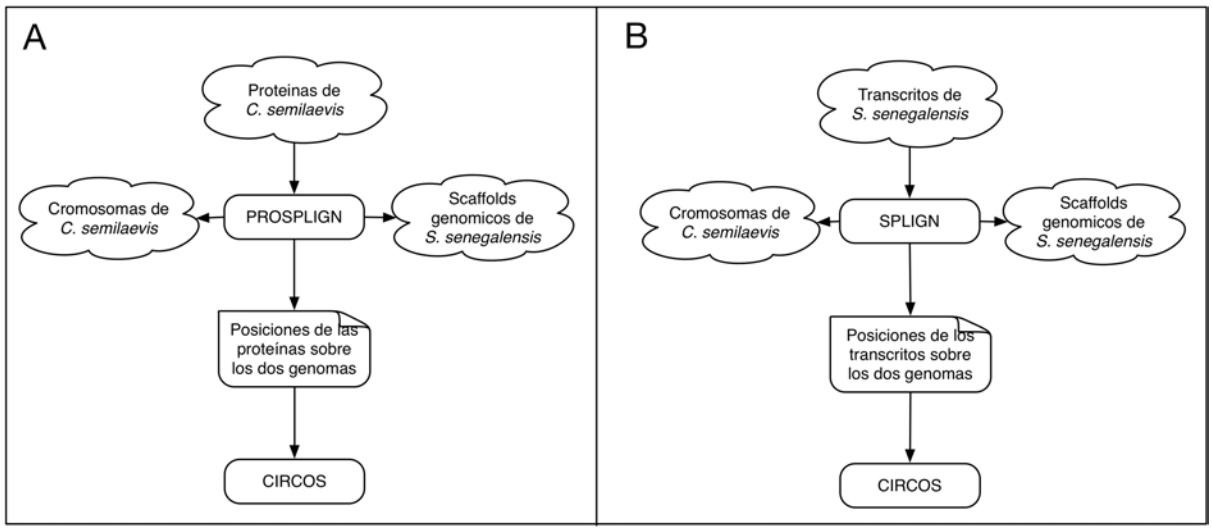

Figura IV.56*: Método utilizado para correlacionar los scaffolds de S. senegalensis con los cromosomas del C. semilaevis. A: posicionando basado en las proteínas de C. semilaevis sobre ambos con el programa PROSPLIGN (http://www.ncbi.nlm.nih.gov/sutils/static/prosplign/prosplign.html). B: posicionando los transcritos del lenguado senegalés sobre ambos con el programa SPLIGN [194]. CIRCOS hace referencia al programa que se usó para visualizar los resultados (véase la figura siguiente)*

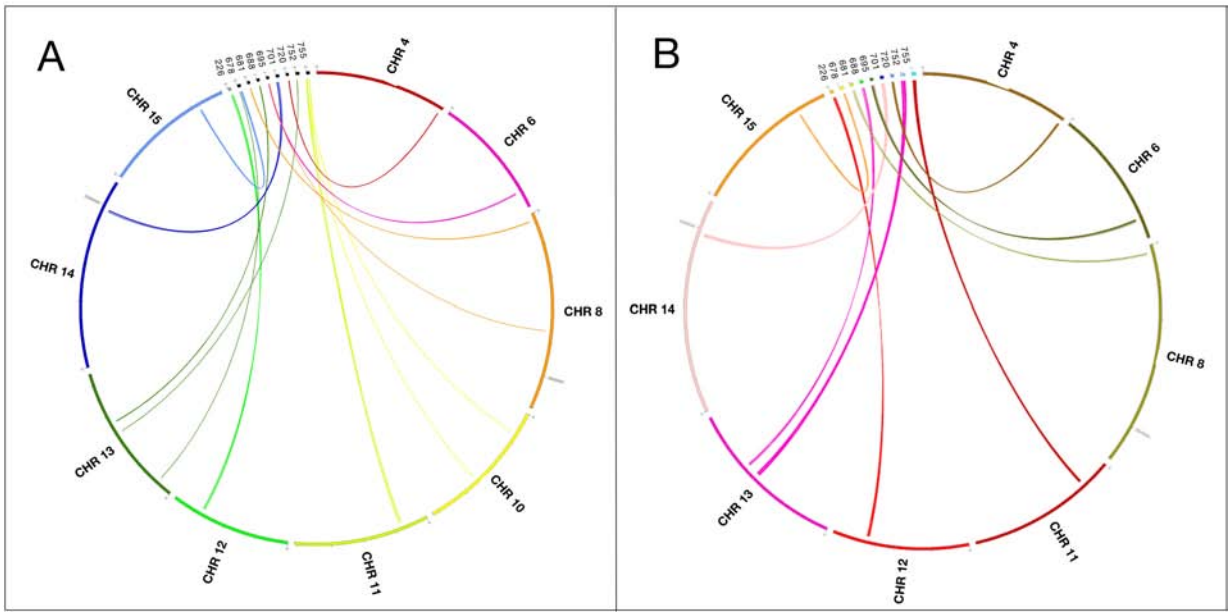

***Figura IV.57**: Visualización de la sintenia de los 9 scaffolds más largos del lenguado senegalés (755, 752, 720, 701, 695, 688, 681, 678 y 228) y los cromosomas publicados del lenguado Cynoglossus semilaevis [160] en función de la correspondencia obtenida con las proteínas de Cynoglossus semilaevis (identidad mínima del 70% en ambas especies (A)), y con transcritos del lenguado senegalés [143] (B). Las coincidencias fueron representadas con Circos [191]*

En la figura IV.58 se muestra un ejemplo de un alineamiento discontinuo pero extenso utilizando el programa GEvo (`https://genomevolution.org/ CoGe/GEvo.pl`) [133] entre el *scaffold* 1145 de *S. senegalensis* y el cromosoma 1 de *C. semilaevis* (los dos daban alineamientos con blast). Se aprecia que los fragmentos alineados tienen el mismo orden entre ambos, de lo que se puede deducir que los dos genomas son colineales. Del mismo modo se aprecia que la zona global alineada está más expandida en el caso de *C. semilaevis,* lo que se podría explicar por las inserciones en el genoma de *C. semilaevis* en esta zona o, por el contrario, deleciones en el genoma de *S. senegalensis* en esta zona ocurridas durante la evolución de estas dos especies.

Estos resultados nos invitaron a clasificar los *scaffolds* del lenguado senegalés en cromosomas en función de la correspondencia con los cromosomas de *Cynoglossus semilaevis*, aunque esta especie contenga cromosomas sexuales que no están definidos en los cariotipos del lenguado senegalés. A estos *scaffolds* ordenados los denominamos *super-scaffolds*.

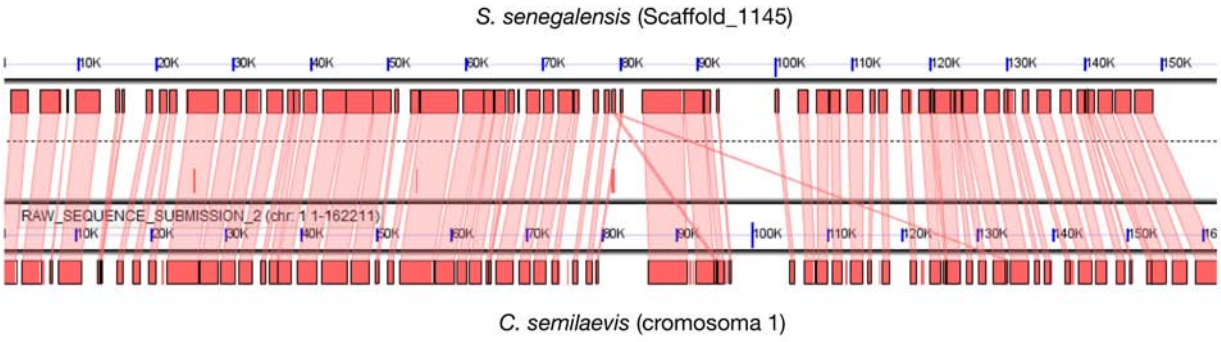

Figura IV.58: *Ejemplo de alineamiento entre el Scaffod 1145 de S. senegalensis y el cromosoma 1 de C. Semilaevis. Las zonas mostradas tienen un tamaño aproximativo de 150 kb. Se nota que fragmentos alineados se encuentran en el mismo orden entre las dos secuencias, lo cual indica la existencia una cierta colinealidad entre ambos genomas*

IV.3.2.4. Los *super-scaffolds* como posibles cromosomas del lenguado senegalés

La colinealidad entre los dos genomas sirvió para organizar los *scaffolds* del lenguado senegalés en *super-scaffolds* en función de los bloques de similitud con los cromosomas de *Cynoglossus semilaevis,* tal como se muestra en la figura IV.59. Como resultado, tan solo 7 971 (23,3%) de los *scaffolds* fueron organizados en *super-scaffolds* mientras que 26 205 fueron descartados (tabla IV.31) principalmente debido a redundancias, solapamiento, o porque no mapeaban sobre los cromosomas de *Cynoglossus semilaevis* (#1, #2 y #3; figura IV.59). Aunque el porcentaje de los *scaffolds* organizados en *super-scaffolds* fue bajo, el tamaño medio resultó ser mucho más alto que aquel de todo el conjunto (58 273 y 17 565 respectivamente) (tabla IV.31). De hecho, la suma total sus bases (464 497 837 pb) conforma una fracción grande y bastante representativa del genoma ensamblado del lenguado senegalés (77,3%). La mayoría de los *scaffolds* descartados eran pequeños (tabla IV.31), aunque algunos puedan contener zonas del genoma o genes propios al lenguado senegalés que no están (o son muy diferentes) en *Cynoglossus semilaevis*.

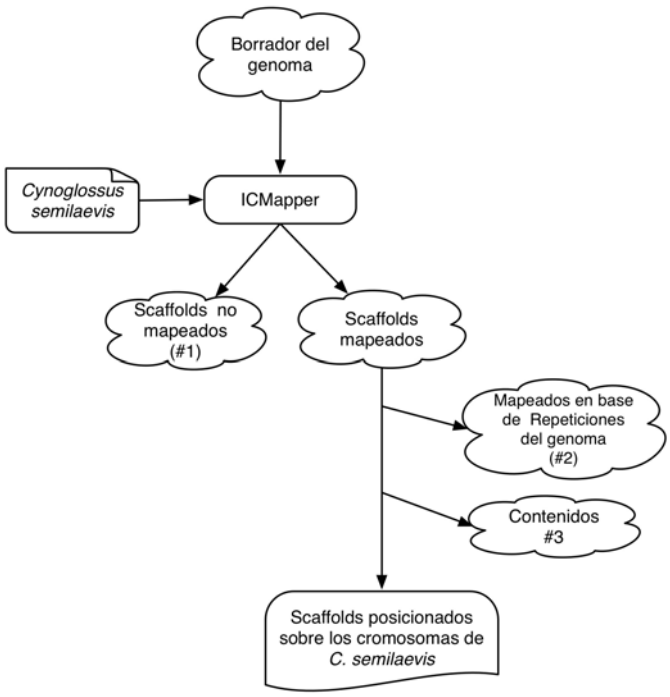

Figura IV.59: *Posicionamiento de los scaffolds de lenguado senegalés sobre los cromosomas de Cynoglossus semilaevis mediante el uso de ICMapper. Se eliminaron los scaffolds que mapearon solo sobre repeticiones. Se conservaron los que tienen el alineamiento más largo o en caso de alineamientos iguales, el scaffold más largo. Después se eliminaron los scaffolds que por su posición estuvieran contenidos dentro de otros más largos*

La distribución de los *scaffolds* del lenguado senegalés en los cromosomas del lenguado *Cynoglossus semilaevis* se presentan en la tabla IV.33, donde se observa que los cromosomas de 1 a 20 y el cromosoma Z estaban bien cubiertos (en todos la cobertura supera el 100%, lo que se podría explicar por el hecho que los datos genéticos indican que el genoma del lenguado senegalés es mayor que el del genoma de *Cynoglossus semilaevis*). Hay que señalar que en el cromosoma W mapearon muchos menos *scaffolds* que los demás cromosomas (figura IV.60) y que además sus alineamientos con los *scaffolds* del lenguado senegalés fueron mucho menores que los de los demás cromosomas (de hecho, el mapeo ocurría principalmente sobre zonas repetidas). Esto nos lleva a pensar que este cromosoma W no está presente en el genoma ensamblado. Esto también concuerda con el hecho de que se ha secuenciado una hembra de lenguado senegalés, y se supone que el cromosoma sexual femenino de la hembra es el Z [160] (que sí está representado en la tabla IV.33). Por eso el lenguado senegalés contiene secuencias del cromosoma Z (determinante del sexo femenino en cariotipos ZZ), pero no del cromosoma W (determinante del sexo masculino en cariotipos WZ). Esta hipótesis se confirmará en un futuro próximo mediante el mapeo de secuencias de macho sobre este cromosoma.

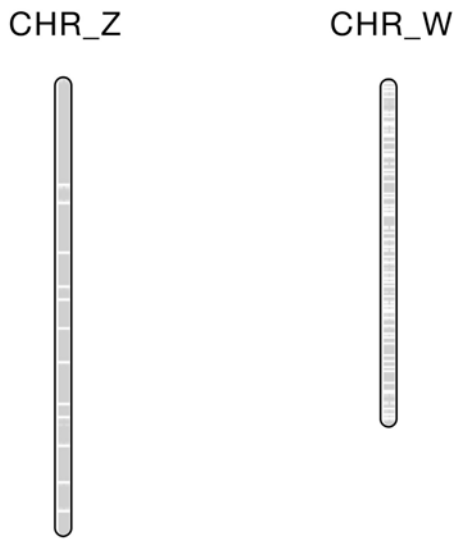

Figura IV.60: *comparación de la cobertura en scaffolds de los cromosomas Z y W del lenguado Cynoglossus semilaevis*

Tabla IV.33: *Distribución de los scaffolds del lenguado senegalés en super-scaffolds sobre la base de los cromosomas de Cynoglossus semilaevis*

Cromosoma de *Cynoglossus semilaevis*	Longitud (nt)	Número de *scaffolds* mapeados	Longitud total de los *scaffolds* mapeados	% cubierto	N50 (nt)
1	34 528 841	658	34 594 364	100	99 530
2	20 052 193	355	22 989 725	115	112 942
3	16 251 686	248	17 823 805	110	137 182
4	20 010 643	332	22 117 000	111	116 619
5	19 279 693	390	21 680 530	112	104 612
6	18 840 743	312	21 278 667	113	119 865
7	13 814 722	260	15 953 794	115	107 816
8	30 151 083	570	31 888 917	106	108 840
9	19 616 557	409	22 150 493	113	91 202
10	21 015 401	354	21 939 736	104	110 738
11	20 528 231	413	22 193 124	108	99 317
12	18 396 836	349	20 005 025	109	122 179
13	21 921 923	417	24 195 529	110	129 954
14	28 847 931	601	29 923 082	104	104 110
15	20 091 810	395	20 507 432	102	116 280
16	18 785 820	341	19 270 135	103	110 670
17	16 464 107	302	18 157 659	110	112 368
18	15 207 555	278	17 440 933	115	114 700
19	17 744 385	314	19 787 260	112	105 947
20	15 231 414	214	17 321 386	114	125 817
Z	21 910 988	459	23 279 241	106	93 798
W	-	-	-	-	-

IV.3.2.5. Validación de los super-scaffolds

Como los *super-scaffolds* se determinaron en función de la similitud de las secuencias de nucleótidos, este resultado necesitaba algún tipo de confirmación. Se realizaron dos diferentes tipos de análisis confirmatorios: localización de los transcritos de lenguado descritos en [143] y la localización de marcadores SSR descritos por otros autores.

IV.3.2.5.1. Localización de los transcritos de lenguado.

Los transcritos del lenguado senegalés se extrajeron de SoleaDB [143] y se localizaron sobre los cromosomas de *Cynoglossus semilaevis* y los *super-scaffolds* del lenguado senegalés con el programa informático SPLIGN [194], tal como se muestra en la figura IV.56-B. También se confirmó que los *scaffolds* descritos en la tabla IV.33 confirmaban su posición en los *super-scaffolds* cuando se mapeaban en ellos los transcritos de lenguado senegalés. Los reultados se presentan en la tabla IV.34, donde entre el 32,0 y el 45,0% de los *scaffolds* incluidos en los *super-scaffolds* contienen transcritos, con la excepción de los cromosomas 20 y Z donde el procentaje subió al 62,1%. Entre 72,2 y 94,0% de los *scaffolds* con transcritos se localizaron en el mismo *super-scaffold* inicialmente asignado (tabla IV.33) con que indica que los *super-scaffolds* representan una buena estrategia para el estudio del genoma del lenguado senegalés. Aun más interesante, el cromosoma Z tiene una cobertura en transcritos similar a los otros cromosomas, pero se observa una correspondencia claramente menor que en los otros cromosomas (40,7%), lo que sugiere que los genes en el cromosoma sexual Z están presentes en el lenguado senegalés, pero quizás distribuidos en otros cromosomas. El hecho que sobre 40% de los *scaffolds* del lenguado senegalés organizados en *super-scaffolds* contienen un transcrito puede indicar que solo este porcentaje de secuencias en los cromosomas son codificantes, lo cual debe ser confirmado cuando el borrador de genoma esté completamente validado y anotado.

Tabla IV.34: *Confirmación de los scaffolds organizados en super-scaffolds en función de la presencia de los transcritos del lenguado senegalés*

Cromosoma de *Cynoglossus semilaevis*	*Scaffolds* del lenguado senegalés que incluyen transcritos	Confirmados (%)	Grupo de ligamiento	SSRs localizados
1	260/658 (39,5)	230/260 (88,5)	LG15	5
2	145/355 (40,8)	130/145 (89,7)	LG8	8
3	116/248 (46,8)	89/116 (76,7)	-	-
4	137/332 (41,3)	105/137 (76,6)	-	-
5	149/390 (38,2)	123/149 (82,6)	LG3+LG26	16 (14+2)
6	128/312 (41,5)	110/128 (85,9)	LG6	5
7	108/260 (41,5)	94/108 (87,0)	LG11	6
8	221/570 (38,8)	188/221 (85,1)	LG7+LG19	5 (3+2)
9	142/409 (34,7)	131/142 (92,3)	LG1*	10
10	139/354 (39,2)	126/139 (90,6)	LG10	5
11	150/413 (36,3)	119/150 (79,3)	LG2	8
12	114/349 (32,7)	106/114 (93,0)	LG12	4
13	149/417 (35,7)	140/149 (94,0)	LG16	3
14	192/601 (32,0)	173/192 (90,1)	LG9+LG18	8 (6+2)
15	145/395 (36,7)	130/145 (89,7)	LG13+LG23	7 (5+2)
16	118/341 (34,6)	106/118 (89,8)	LG17+LG25	5 (3+2)
17	136/302 (45,0)	127/136 (93,4)	LG1*+LG14	6 (4+2)
18	91/278 (32,7)	80/91 (87,9)	LG5	2
19	136/314 (43,3)	119/136 (87,5)	-	-
20	133/214 (62,1)	96/133 (72,2)	LG21+LG27	4 (2+2)
Z	283/459 (62,7)	116/283 (41,0)	LG4	6
W	- -	- -	-	-
-			LG20	0
-			LG22	0
-			LG24	0

*LG: grupo de ligamiento. *LG1 contiene SSRs que mapean en dos super-scaffolds diferentes. Los grupos de ligamiento se tomaron de [195]*

IV.3.2.5.2.Validación de los marcadores moleculares del lenguado senegalés

La primera validación con marcadores moleculares se realizó sobre la base de una versión preliminar del ensamblaje del genoma que estaba formada por 421 497 *scaffolds*. Después de posicionar estos *scaffolds* sobre los cromosomas de *C. semilaevis* utilizando ICMapper, se seleccionó un conjunto de 665 *scaffolds* que tenían el criterio de ser largos (>20kb) y de tener un cluster de alineamiento con el cromosoma de *C. semilaevis* (véase el algoritmo de ICMapper en el apartado IV.1.4) que cubre más del 90% de la longitud del

scaffold. Estos *scaffolds* fueron ordenados según su posición sobre los cromosomas de *C. semilaevis* y enviados al grupo de investigación de IFAPA EL Toruño (Cádiz), el cual seleccionó a partir ellos un grupo más pequeño de *scaffolds* con secuencias de SSR que estaban bien repartidos sobre los cromosomas de *C. semilaevis*; posteriormente se diseñaron parejas de cebadores para ser utilizadas en un experimento de amplificación de los marcadores SSR con PCR multiplex. El experimento fue exitoso ya que todos los marcadores SSR se amplificaron correctamente. Una vez acabado el experimento, se nos remitió la lista con los *scaffolds* y cebadores utilizados. Para conocer las posiciones de los marcadores SSR en el nuevo ensamblaje, a los nuevos *scaffolds* (posicionados sobre el cromosoma de *C. semilaevis*; tabla IV.31) les alineamos las parejas de cebadores utilizados en el experimento de SSR con blastn (comprobando que los dos cebadores tienen un alineamiento exacto y con una distancia corta). De los 113 parejas de cebadores utilizados en el experimento, 88 se posicionaron en el mismo cromosoma que antes (al cual pertenecía el *scaffolds* del cual proceden), mientras que a las otras 25 se les asignó un cromosoma diferente. Como los *scaffolds* nuevos eran más largos y cubrían una zona más amplia del genoma de *C. semilaevis* supusimos que sus posiciones en los cromosomas eran más seguras, por lo tanto, consideramos que eran más fiables estas nuevas asignaciones de los marcadores SSR a los cromosomas. No obstante, 13 de estas 25 nuevas asignaciones de marcadores SSR a los cromosomas resultaban dudosas, puesto que los scaffolds de donde procedían los cebadores alineaban con dos o varios *scaffolds* nuevos que pertenecían a cromosomas distintos, lo cual nos hizo sospechar que se formaron con ensamblajes erróneos. En la tabla IV.35 se presenta la lista con los marcadores SSR, los cebadores utilizados en la amplificación PCR multiplex, así como los nuevos *scaffolds* donde mapearon estos cebadores.

En una segunda validación, se utilizaron los marcadores SSR previamente descritos en la bibliografía [195, 196]. De los 129 marcadores SSR (o sus cebadores cuando la secuencia no estuvo disponible), la secuencia de 113 se pudo localizar sobre los *scaffolds* del lenguado senegalés (Tabla 4.21) para confirmar la asignación de los *scaffolds* a los *super-scaffolds*. Solo 10 de los 113 SSR presentaron una localización dudosa (no incluidos en la tabla 4.21) y 6 no fueron localizados, que correspondían a los grupos de ligamiento LG20, LG22 y LG24. Por otra parte, ningún marcador SSR fue localizado en los cromosomas 3, 4 ni 19. Como se esperaba, el cromosoma W no contenía marcadores SSR. La localización de los SSR sirvió también para correlacionar los *super-scaffolds* (y por lo tanto los cromosomas de *Cynoglossus semilaevis*) con los grupos de ligamiento lenguado senegalés (Tabla 4.21, dos ultimas columnas). También se observa que el grupo de ligamiento 1 (LG1) esta distribuido entre dos diferentes *super-scaffolds* (marcados con asteriscos en la tabla 4.21), lo que indica que, a pesar de la muy alta sintenia entre ambos lenguados, los cromosomas reales podrían ser ligeramente diferentes. Es de destacar que el cromosoma de determinación de sexo femenino Z tiene poca cobertura (mientras que 62,1% de los *scaffolds* contienen trascritos, solo 40,7% están localizados en este cromosoma). Esto puede indicar que el cromosoma de determinación de sexo femenino en *Cynoglossus semilaevis* no tiene una buena correspondencia en el lenguado senegalés. Estos resultados, y los presentados en el apartado anterior, sobre los cromosomas W y Z explican la dificultad para determinar los cromosomas sexuales del lenguado senegalés.

La alta sintenia entre el lenguado senegalés y el lenguado *Cynoglossus semilaevis,* y la alta cobertura de los *scaffolds* del lenguado senegalés sobre los cromosomas del lenguado *Cynoglossus semilaevis* permitió crear el mapa genético del lenguado senegalés ilustrado en la figura IV.61. Por cada cromosoma del mapa genético se situaron de una parte los marcadores SSR nuevos indicados en la tabla IV.35 y de otra parte los marcadores SSR previamente descritos en [195, 196]. Se incluyeron también los grupos de ligamiento del lenguado senegalés descritos en [195] correspondientes a cada cromosoma tal como se indicó en la tabla IV.34.

Tabla IV.35: *Marcadores SSR desarrollados en este trabajo sobre los scaffolds del lenguado senegalés y su amplificación con PCR multiplex en el ADN genómico*
(*) posición dudosa del marcador SSR

Scaffold	CHR	Motivo más importante en la secuencia de referencia	Nombre de cebadores	Cebador directo y reverso (5'-->3')
Scaffold_3268 (*)	1	(TTTA)10	SSeneg13367	D: CGCTCCCTGGATTACAGTTC
				R: gtttACGGCGAGATACGATCACA
Scaffold_3627	1	(GATG)6	SSeneg4306	D: CCGTTCAGCTGTTGAGGATT
				R: gtttAGTGTTTCTGCCCCCTCTCT
Scaffold_760	1	(GGAT)6	SSeneg2307	D: GGGTAAGAGATGACTGCAGGA
				R: gttTCTGGCATCCTTGAATGAAA
Scaffold_1072	1	(ATAG)13	SSeneg8782	D: AAGCACCTAACGGAATCTGC
				R: gTGTCTGACCGACATTTTGGA
Scaffold_12926 (*)	1	(CTGT)7	SSeneg287	D: CGAGTGCCTTCACAATACCTG
				R: gtttCAGAGGAAATGGCAAGCAG
Scaffold_2471	1	(CCAT)8	SSeneg5412	D: TCTCAGCTGTCCAAAGAGCA
				R: gttTACTGGGTGGTAGCGGAAAG
Scaffold_22268 (*)	1	(ACAG)7	SSeneg4382	D: GAGGGACATGACGATCACCT
				R: gttTGTTTAAAAGCTGCGCTGTG
Scaffold_1721	1	(AGAT)14	SSeneg5891	D: CAAACACACGACACTACAGCAA
				R: gCACAGCAGTTTCATTTCTCTGC
Scaffold_21912	1	(TTTA)9	SSeneg6689	D: TGATTGGCAACTTCAGGAGA
				R: gtttCACTTTCCATTCCCCAGTGT
Scaffold_3043	1	(ATTG)10	SSeneg4328	D: CTCTGCCATCTACACCAGCA
				R: gtttAGTGGAGGGAGACTGAGCAA
Scaffold_2626	2	(TCTT)9	SSeneg16050	D: CGCTCGTTGTCATTGCTTTA
				R: gtttCGCCTCTTTATGGACTCAGC
Scaffold_3788	2	(AATG)8	SSeneg12054	D: TGTCTTTTTGGCATTGGATG
				R: gttTCGTGCCACAACAAGAAAAG
Scaffold_787	2	(ATCT)23	SSeneg12678	D: CCGGAAGCAAAACAAAGGTA
				R: gtttATCGACTGTTGGCCTTTTGT
Scaffold_730	2	(CAGT)7	SSeneg1505	D: CCACAACACAACCACACACTC
				R: GGTGTGGCCTCTACACTCCT
Scaffold_855	3	(TTTA)9	SSeneg4374	D: AGCATGGACGTTGAATGTGA
				R: gtttCACACGGAACGATGACAAAG
Scaffold_601	3	(AATA)9	SSeneg7919	D: CAGCCAGACGTTAAAATGAACA
				R: gTACCTGAAGGCGCTCTGATT
Scaffold_4719	3	(TTTC)7a(TTCT)5	SSeneg10667	D: CGCGTTGGGAAGATAACATT
				R: GTGCCAAGAAAATCCAGCAT
Scaffold_2970	3	(AGTG)9	SSeneg5850	D: CAGAGAGCGATGACACATTGA
				R: gtttATTCAAACGCCGTCTTCATC
Scaffold_1976	3	(GGAT)8(AGAT)24	SSeneg4003	D: AAGGCTGAATCATGTCCTCAA
				R: GAGTGCGCCATCCATAAAAT
Scaffold_4327	3	(AGAGA)12	SSeneg14597	D: ATTGTTGTTGGCCTCCACTC
				R: gtttCATAGGGGTCGGTTTGAGAA
Scaffold_5580	4	(GTGA)9	SSeneg12137	D: ACTCACGGTTCGTGGAGAAG
				R: gTCAAATGGAGGCTTGTGGA
Scaffold_1562	4	(AGAT)20	SSeneg9009	D: AAGCATGACATTAAAATGGCTAAAG
				R: gtttATAGGGCCCCCATAAAGCTA
Scaffold_2448	4	(TATTT)16	SSeneg12624	D: ACTCAAGTGGATTCGCCATC
				R: gTCCTGTTACTGAAGGTAGGTGCT
Scaffold_1282	4	(TCTT)8	SSeneg12095	D: ATGGCTCAGACCCACCATAC
				R: gttTGAAGAATGGGGCAAAAATC
Scaffold_402	4	(GTGC)8	SSeneg5713	D: TCCTCATGGAGACCAGAACC
				R: gtttCATACCAGGAACCCTCTGGA
Scaffold_2128	4	(AATG)8	SSeneg45	D: ACAAACAGCCCGTAGGAATG
				R: gTCCCCCAACACTCTTGGATA
Scaffold_981 (*)	4	(TGAA)16	SSeneg3978	D: AAAGCGCTAAATCGCACACT
				R: gTTTGATGCAGAGGCAGACAC

Tabla IV.35: *Continuación*

Scaffold	CHR	Motivo más importante en la secuencia de referencia	Nombre de cebadores	Cebador directo y reverso (5'–>3')
Scaffold_897	5	(GATA)17	SSeneg11269	D: AGAGGCGGAAGCAACTGAG R: gTTTTGTGGGCTGTGGTGTAA
Scaffold_193	5	(TAGA)6	SSeneg3502	D: GTGTTACCACTTTACAATGGCAGT R: gtttAGGTTTGCAGGATTTGGAGA
Scaffold_9270 (*)	5	(GTGA)7	SSeneg12300	D: TCCTTCACAGCATTGAGTCG R: GCTGCCAAACAGGAGAACAT
Scaffold_3830	5	(AGAC)10	SSeneg13116	D: TCGTGGGGGTAAGGGTAAGT R: gCCCTCTGTTATGGAGCGTGT
Scaffold_1338	5	(CTTT)10	SSeneg5827	D: GACGTGTGACGTTTGTCCAG R: gTTTGATTTCTGAATGCTGTTCAT
Scaffold_3528	6	(TTCA)6	SSeneg4572	D: CATCCTCCACTTCCCACACT R: gTCCTCTGGTGGAGCAAAATC
Scaffold_706	6	(ATAG)10	SSeneg3069	D: GCATGCAGGGCCACTTTA R: gtttCCTGCTGCCTCAATATCGTT
Scaffold_672	6	(TAGA)13	SSeneg2487	D: GCAAAAGCATCACATTACATTTTG R: gttTCAATCACCATGTCTTGCCTA
Scaffold_6439	6	(CATT)8caatcaatcaa(TCAT)6	SSeneg53551	D: TTTCATGGTCAAACGTCACAA R: gTATGCGATGGAGTGCAAAAG
Scaffold_1159	7	(ATTAA)8	SSeneg395	D: TTTCCCATGTCATCTCAGCA R: GTTCTCCCCGTGTGTGCTAT
Scaffold_6291	7	(AGTG)9	SSeneg14931	D: CCCTCTTATGGAGGACAAAGC R: GTGGATAAACGGGGTTCCTT
Scaffold_1115	7	(GATA)6gatggatagatggatagata gatagata(GATG)11(GATA)21	SSeneg12220	D: AAAGCCATGCCATTTACTGC R: GTATAATGTATGCTATACGAAGTTATTACGA
Scaffold_966	7	(GTGC)5	SSeneg2473	D: AGTGCGGCGATATTTGTTGT R: GCCTCATAAGAACGCCTCTG
Scaffold_956	7	(ACAA)7	SSeneg5899	D: CTGGAGGAGACAGGATCGTT R: gttTGATGAACCAAATGCAGGAA
Scaffold_4544	8	(GATA)11	SSeneg5202	D: TTAAGGTTAGTTATTGGTGACACCTC R: GGACACAAGCCCTACGACAT
Scaffold_456	8	(CCCT)6	SSeneg2894	D: CAGTCCAGTGGCTTGTCAAA GGATCCCAATTTGGTTTCAA
Scaffold_241	8	(CTCA)8	SSeneg5828	D: CGTCTGGAAAATAGGGTGGA R: gTTTCTCCCAGCAGAAGCTGT
Scaffold_21831	8	(TCAT)6	SSeneg1411	D: CCTGCTCCTTACTTTTGACGA R: gCTGCTGTTAAACAACTTCTGCAA
Scaffold_1727	8	(GAAA)8	SSeneg10524	D: TGCTGCTCAGCAAATTGAAG R: gttTCTTTTCTCCTCGCGTTGTC
Scaffold_102	9	(GATA)12	SSeneg6381	D: ACCCAAAACACTGGAAGTCG R: gttTGTAAAACATTAACACATGCAGGA
Scaffold_21792	9	(TAAT)9	SSeneg14542	D: TGTGCACCAGAGAAGGTGAG R: gtttCCTCCATCTACAGCTCCTAATGA
Scaffold_21944 (*)	9	(TATC)19	SSeneg3985	D: TGAATTTCCTTCGGGATCAA R: gtttACTGAGTTCCACATGTGTCCA
Scaffold_5545	9	(GATA)8	SSeneg1669	D: CAGGAGTGCCACTGAAATGTT R: gTGTGAATTTCTCTGTGAGATGAATAG
Scaffold_1195	9	(TCTA)12	SSeneg774	D: TGGTCTCCAGTTGCACACAT R: gttTGTTTCCCAACACACGTCAT
Scaffold_1861	9	(TAGA)15	SSeneg2996	D: CCATAAGTATTATCTTAAATAGGATCCAGA R: gAGTGCCCATTACACACAGCA
Scaffold_1476	9	(CTAT)13	SSeneg387243	D: TGTGGGAGGGTTGTGAAAAT R: gtttACGTATAGGGCCAGGGAGAG
Scaffold_36	10	(TCTG)7	SSeneg16258	D: TTGCAACGCTTTACCAACTG R: GCCTCCTCCTCTGCCTGT

Tabla IV.35: *Continuación*

Scaffold	CHR	Motivo más importante en la secuencia de referencia	Nombre de cebadores	Cebador directo y reverso (5'-->3')
Scaffold_744	10	(GTCT)7	SSeneg433	D: TAAGCCACTGAAATGTCAAGC
				R: gttTCAGTACAGCCGCTGATGG
Scaffold_14518	10	(GGAG)7	SSeneg14333	D: TTAACGGCATCCAGATGTGA
				R: gtttATGGAAATGGAGGGTCTGTG
Scaffold_2812	11	(ACAG)10	SSeneg11209	D: CTACATGTGCTTACGCTGCAC
				R: gATTAAGCACCTGGACGGATG
Scaffold_1209	11	(TAGA)12	SSeneg10308	D: TCACAGCAGCAATCAAGACA
				R: gtCAGAACATCTGTTCCCTCTGTATC
Scaffold_222	11	(GACA)7	SSeneg437	D: TGGAGGGCTCAAAAGAGAGA
				R: gCATAGCTCAGCGAGGAAACC
Scaffold_3709	11	(TAAA)11	SSeneg106	D: TGGCACATCCTGTGCATT
				R: gttCACTCAGGATTTGACAGAGGATT
Scaffold_272	11	(TAGA)18	SSeneg4039	D: TGGGTTTGAAAGAGAATTCCA
				R: GGTTGCTTTGCAAAAGACAGT
Scaffold_3073	12	(ATCT)6	SSeneg3041	D: CCCAATAATCTGCACAGCAA
				R: GCGAGAAAGAGAAAGTCATGAGA
Scaffold_765	12	(TCCA)7	SSeneg827	D: CCATACCCAGGCTGAATTTAC
				R: gCATGTGGAGGATAAAGTGGTAGA
Scaffold_21931 (*)	12	(AAAG)12	SSeneg4065	D: CTGTCCGTGACATCACACCT
				R: gtttACCATTCCCATTGCCAGTAG
Scaffold_884	12	(GGAT)8	SSeneg3415	D: ACCGCTGGGATGTACTGAAG
				R: GCCTGTCCATTGTGAGGAAT
Scaffold_228	12	(CTAT)12	SSeneg3342	D: AATATGAGCAACTCTAAGGGAATTG
				R: gtttCAGAGATCGGTAGATAGGAATTGTT
Scaffold_21911	13	(TAAAA)5	SSeneg6982	D: TTGAATACACATTGGCGTCAC
				R: gttAATGCACTACCCGCTGTTTC
Scaffold_1113	13	(AGAA)6	SSeneg87	D: GCCTCTGTGGCAGAGCTATT
				R: gttTGGCTTTCTGTTTGATATTGTCC
Scaffold_403	13	(GCAG)7	SSeneg854	D: ACGTCCCTCACAAATTCTGC
				R: GTGTCTCCTCCCAGCTTCC
Scaffold_21965 (*)	13	(TAGA)11	SSeneg15332	D: AAGAGCTGATGGTGCAAGGT
				R: gTCCTTAAAGCAAGTGAAAAACCA
Scaffold_2167	13	(GATA)18	SSeneg4081	D: GCCTCGTGATTGAGCTGAG
				R: gtttCTCCTGAATCACCCACTGGT
Scaffold_987 (*)	13	(TTTC)16	SSeneg1147	D: CGGGCACAGTATTCCTCTTT
				R: gttTCAAGCAAAATTCTTGATGTCAG
Scaffold_700	14	(TATT)8	SSeneg17673	D: AGTGGACTTTTTGTTGGGGTTA
				R: gtttACCACATGGACAGAATGCAC
Scaffold_1311 (*)	14	(CTGT)8	SSeneg5919	D: AACAGGCTCCATGGATTCAC
				R: gTCAGTGGTTTGCACCGAATA
Scaffold_886	14	(CATA)9	SSeneg348796	D: TTACTTTCTGCTGTGCTAGTGTGT
				R: gTTAGCAGGTGTACCTAAATAAGTGG
Scaffold_689	14	(TATC)12	SSeneg10877	D: TTTCAGCATTTTACGGCTCA
				R: gttTTTGCTGTTGTCAGCCTCTG
Scaffold_3676	15	(TCTT)12	SSeneg6326	D: GCTGAACAAACACCCCATTC
				R: gtCAACCCGGCACAATTAGTCT
Scaffold_8254 (*)	15	(TGGA)9	SSeneg544	D: ATGATGCTGCAGAAATGACG
				R: gCCCCCACTGTGGCACTAATA
Scaffold_3649	15	(TCTG)6	SSeneg17159	D: TTAATGCCAAGTGGGTGTTG
				R: gtttACTGAGAGCGTGTCAGTGGA
Scaffold_1753	15	(GATA)12	SSeneg9042	D: TGTTGCCAACTAGCTTAGATAAACA
				R: GCTATTGAAGCTACTGTAGTGAGGAA
Scaffold_131	15	(TGTT)7	SSeneg1723	D: CAATGGAGCAGAGGTGCTTT
				R: gTTCATAGTGCCATAACAATGTGG

Tabla IV.35: *Continuación*

Scaffold	CHR	Motivo más importante en la secuencia de referencia	Nombre de cebadores	Cebador directo y reverso (5'–>3')
Scaffold_1294	16	(CTAT)8cta(CTAT)18	SSeneg1973	D: CATAGGTCATTAGACTTAATGTATCTGAA
				R: gtttCCAACTGTTTTTCGCACAGA
Scaffold_22147	16	(TAGA)9	SSeneg2891	D: TGGTCACAGGAAAAAGCTCA
				R: GCCAAATCAAAGAACCAATCA
Scaffold_3405	16	(TTAT)8	SSeneg12417	D: GTTAACACTGCGGCCACTTT
				R: gtttACCAGCATTTGTCAACCACA
Scaffold_1941	16	(TTTA)6	SSeneg4608	D: ACAAATAATGCTGCTACGTGGA
				R: gtTTGTAGACCCATGTCACTCACC
Scaffold_677	16	(AAGA)11	SSeneg2083	D: GGAGGACATGGAAGGAAACA
				R: GCTGTTCTCTGGAGGCAAAC
Scaffold_21790	17	(AAAT)11	SSeneg3683	D: TTGGCCAGGTCATCACTGTA
				R: gttTCAAAACATTTGGCTCATTCTG
Scaffold_3596	17	(CAGA)7	SSeneg8412	D: TGGTGTATGGAGGCTCTTCTC
				R: GCATGTAGGACTGTGCGTGA
Scaffold_1383	17	(TCAT)7	SSeneg6827	D: ATTTGTGCTGCCATGTTTCTAC
				R: gtttCCTCATGTGGAGGATAAAGCA
Scaffold_703	17	(CTTT)9	SSeneg247	D: CCGGAAAAGTCACTCACTATTAGC
				R: gtttAGACAGGGTCGATGTTGGTC
Scaffold_885	18	(GAAA)6	SSeneg162554	D: GTGGTGAGATGTATGGAGTAAAGACT
				R: GCCTTTGTGTGTTTTTGTATTCTC
Scaffold_806	18	(ATAC)15	SSeneg1667	D: GAAGTGAACGACCATGTGTGA
				R: gtCAAATCCCTGCAGTCAGGTT
Scaffold_3591	18	(AATC)9	SSeneg2868	D: TATCGCTCACATGCTTCAGG
				R: gtTAAAACCGCATTGTGTGCAT
Scaffold_408	18	(CACT)10	SSeneg7074	D: TCATCGGCTAATCACATCCA
				R: gtttCACCGACACTTTGAACCTGA
Scaffold_2408	19	(TTTC)7	SSeneg11316	D: GCAGCATGTGCTTGAGAAAG
				R: gTTTGCTCCCTCGGAGAATAAT
Scaffold_1766	19	(AGAC)13	SSeneg4083	D: ACTGTGGCAATGGAAACACA
				R: gtttACGATGCCTCATTTTGATCC
Scaffold_21973	19	(GTCT)9	SSeneg566	D: TCAACAAACCTTGCAGCATT
				R: gtttAAACATCCCACAGCACAACA
Scaffold_769	19	(AGAT)10	SSeneg7666	D: TGGGCAGGAAGTCAGCTAAA
				R: gtttCTGTGAACCCAGGTTTCCTT
Scaffold_1082	20	(GAGGA)9	SSeneg1201	D: GAAACAATCCCTCCCCTCAT
				R: gAGCAGCTGGCTGTGAGAGTT
Scaffold_3125	20	(GATA)23	SSeneg5346	D: ACACAAACTCCCTCGCTGTT
				R: gCTGCAGTATCCACCATCTGC
Scaffold_763 (*)	20	(ACAG)7	SSeneg3077	D: TCAGCTCCTGTGCAGCAA
				R: gtAAGCACACGTGTCACATTCC
Scaffold_146	20	(AGAT)24	SSeneg6876	D: TCCTTGGCAAGAACTGACCT
				R: GGACAGCGCAATCCTTTCT
Scaffold_5165	20	(ATCA)7	SSeneg506	D: TTGCTACGATGTGGTCCTTTC
				R: gtttAGCAGGTAACTGGGTTTCCA
Scaffold_1425	Z	(TTAT)6	SSeneg5772	D: TGCCATTCACTCACACTTGC
				R: gATGCCTGGTCCCTATTTTAGAT
Scaffold_22156	Z	(TGAG)15	SSeneg977	D: TCCTCCCACAGTCCAAAAAC
				R: gttCAGCGGACATAGGGAGAAAG
Scaffold_2958	Z	(ATCT)8	SSeneg585	D: AACCATGAAATATTGCTCTTCCA
				R: gCCATTTCACAGAAACGGATGT
Scaffold_1171	Z	(TCTT)11	SSeneg73	D: ACGAGCGCAAAGGAACTTTA
				R: gttAGCTGACATCGCATGACATT
Scaffold_22094	Z	(AGAT)16	SSeneg10804	D: ACCAACATGGCAGAAGAACC
				R: gtTACAAAGAACCAGCCCCAAC

Tabla IV.35: *Continuación*

Scaffold	CHR	Motivo más importante en la secuencia de referencia	Nombre de cebadores	Cebador directo y reverso (5'–>3')
Scaffold_22269 (*)	Z	(GTGC)9	SSeneg90	D: CCTGTGTGCATGTGTGTTCA R: GCATTCAATTCGCTTCACAA
Scaffold_2077	Z	(TCAT)6	SSeneg2596	D: TTCATTTGATTCCTTGTGCTTC R: gtTTGATCAATAACAGTTTAGCAGGTCT
Scaffold_2047	Z	(ATCT)17	SSeneg7987	D: TTGAGACAGATTTTTAATGCAGGA R: gtTTAGCAACGTCCCCTGAAAC
Scaffold_22131	Z	(TGAA)7	SSeneg171	D: GGAGGAAGACCCGACCTAAC R: gTGCAGTTGGACACTTTCTGC
Scaffold_1116	Z	(TGAT)6	SSeneg1988	D: ATCACCACTGGGTTCCCTTT R: gttATTTGGGCAAGGGCATATTT

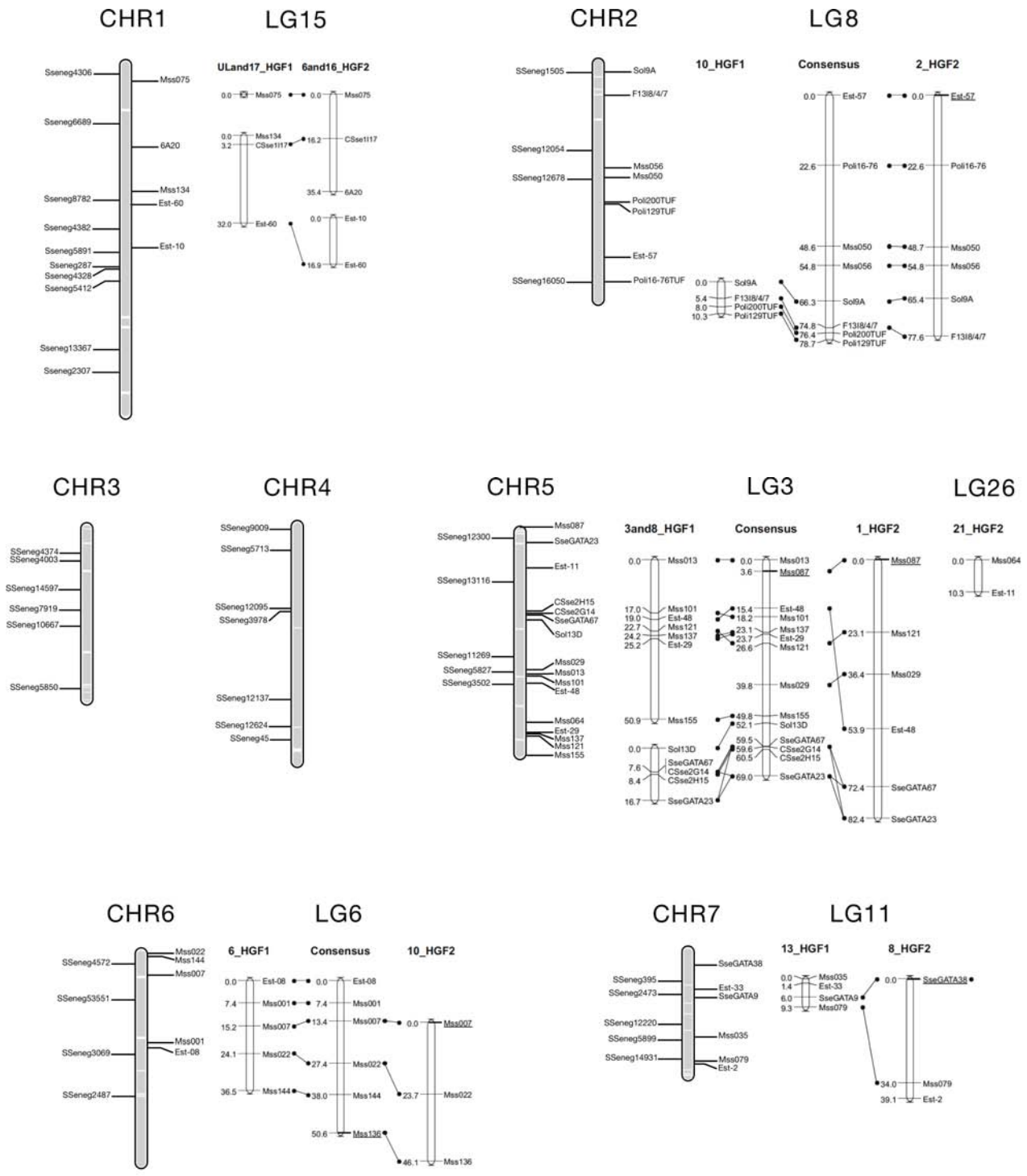

Figura IV.61: *Mapa genético del lenguado senegalés (S. senegalensis) basado sobre la sintenia con los cromosomas de Cynoglossus semilaevis. Por cada cromosoma de Cynoglossus semilaevis se muestra (i) la cobertura de los scaffolds (en gris), (ii) posicionamiento sobre el cromosoma de los marcadores SSR descritos en la tabla IV.35 (a la izquierda), (iii) posicionamiento (a la derecha) de los marcadores SSR previamente descritos en [195, 196], (iv) grupos de ligamiento del lenguado senegalés tal como se describieron en [195] y la posición de los marcadores SSR sobre ellos*

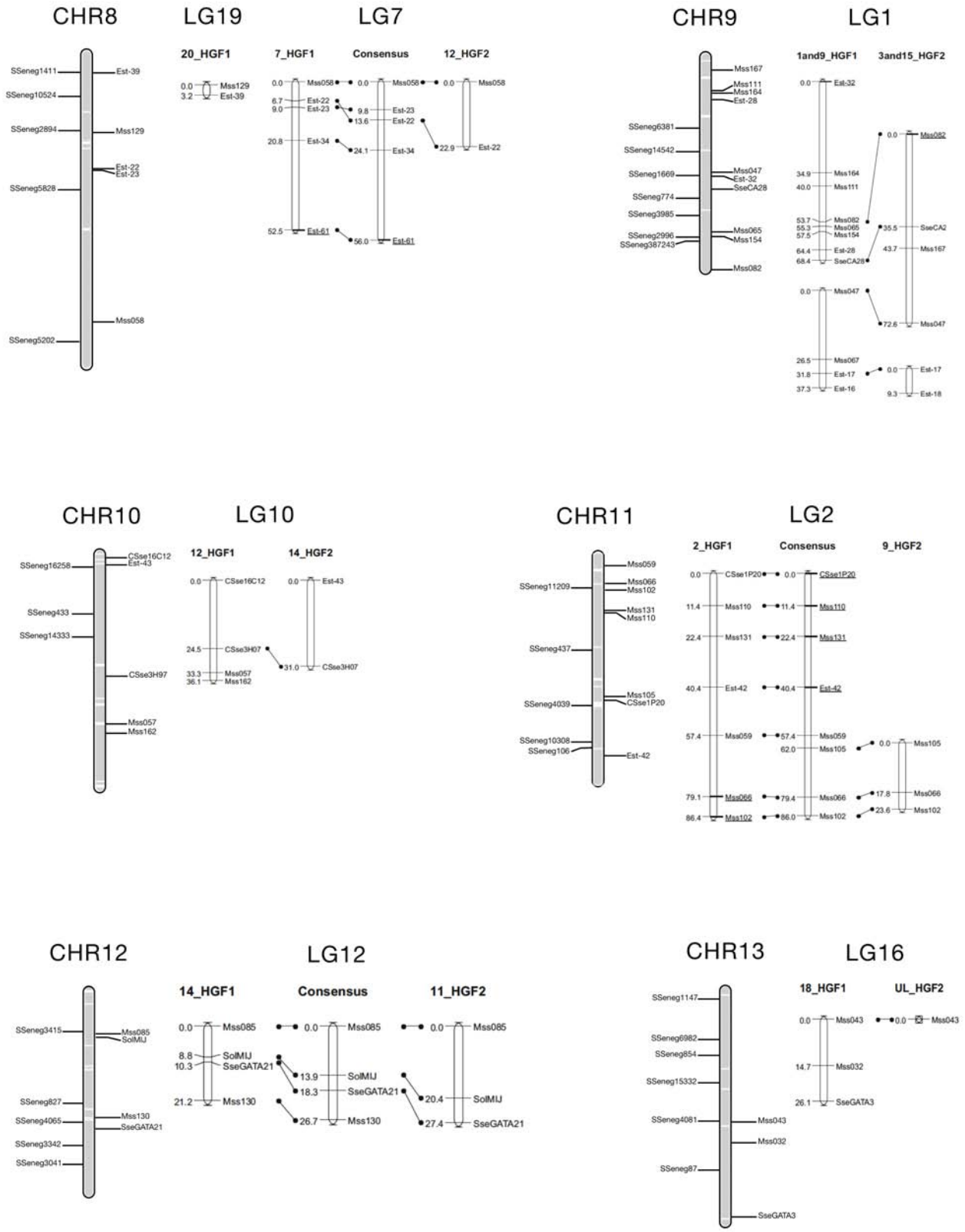

Figura IV.61: *Continuación*

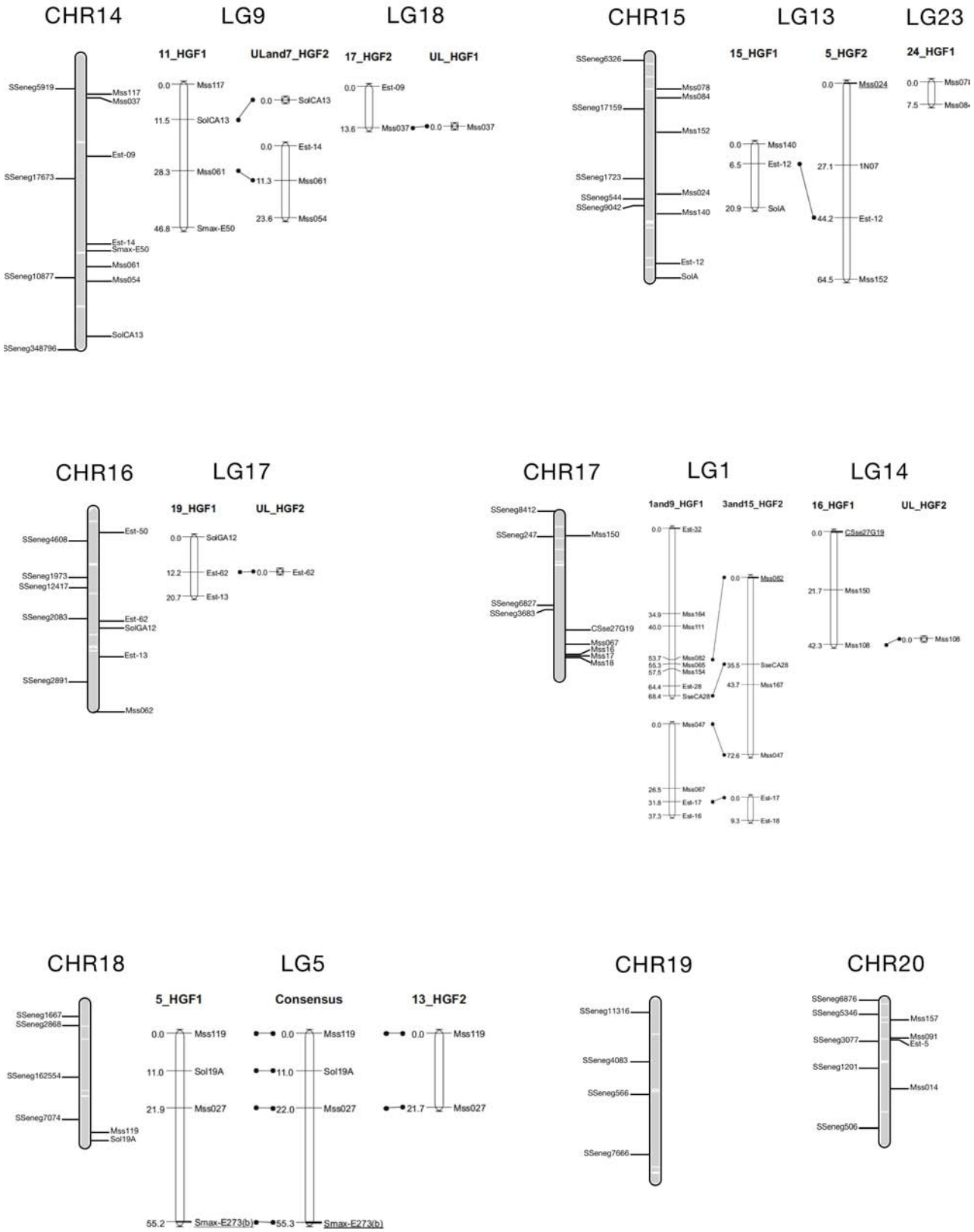

Figura IV.61: *Continuación*

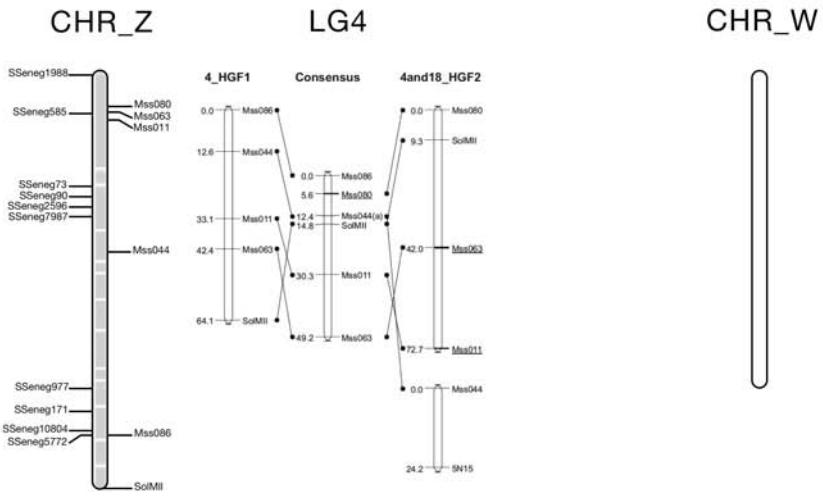

Figura IV.61: *Continuación*

Parte V
Conclusiones

V. Conclusiones

1. Se han definido métodos de comparación de ensambladores de lecturas largas que han permitido destacar como mejores ensambladores MIRA y EULER en transcriptomica y MIRA y CABOG en genómica. Estos métodos se pueden aplicar para probar y comparar nuevos ensambladores.

2. Cominer desarrollado para la detección y corrección automática de errores en ensamblajes de lecturas largas, ha permitido mejorar la calidad de los contigs transcriptomicos y genomicos.

3. ICMapper desarrollado para ordenar contigs y scaffolds basándose en una referencia; ha demostrado ser más preciso y rápido que otros programas del mismo tipo. Resulta muy útil para el acabado de genomas tanto bacterianos como de eucariotas.

4. Se ha optimizado un flujo de trabajo de ensamblaje combinatorio de lecturas largas y cortas para permitir la máxima cobertura de genes. Este flujo de ensamblaje se completa con otro de anotación adaptado a grandes conjuntos de transcritos.

5. Los transcriptomas de *Solea senegalensis* y *Solea solea* representan una fuente de información valiosa para la mejora genética de estas dos especies de gran importancia económica. El análisis de ortología entre las dos especies permitió definir los transcritos ortólogos entre ellas (>97%), los transcritos específicos de especie, los específicos de linaje y los específicos del pez plano.

6. El flujo de trabajo utilizado en *Solea senegalensis* ha servido para ensamblar y anotar los transcriptomas de *Tisochrysis lueta* y *Ruditapes decussatus* lo cual ha ofrecido otra información valiosa que puede ser explotada en los programas de mejora genética de estas especies.

7. Se han realizado mejoras sobre el modelo de la base de datos SustainPineDB para facilitar el acceso a los datos almacenados y ofrecer información adicional a los usuarios. El modelo mejorado se ha utilizado para almacenar los datos de los transcriptomas ensamblados en esta tesis y se está utilizando en el laboratorio para el transcriptoma de otras especies.

8. El ensamblaje y la anotación del genoma de dos cepas de *Photobacterium damselae* subsp. *piscicida* L091106-03H y DI21 ha puesto en evidencia la alta similitud entre ambas; con ello se facilitará el estudio de los factores de virulencia al fin de desarrollar métodos más eficaces de lucha contra ellas.

9. Se ha presentado el primer borrador de genoma de *Solea senegalenses* formado por 34 176 scaffolds siguiendo una estrategia que combina diferentes fases de ensamblaje de lecturas y unificación de los contigs y scaffolds. Gracias a la sintenia con C. semilaevis se han podido organizar los scaffolds de *Solea senegalensis* en *super-scaffolds* equivalentes a cromosomas y se le ha creado un mapa genético que incluye marcadores SSR específicos de esta especie.

Parte VI
Bibliografía

VI. Bibliografía

1. Chang C, Meyerowitz EM: Molecular cloning and DNA sequence of the Arabidopsis thaliana alcohol dehydrogenase gene. Proceedings of the National Academy of Sciences of the United States of America 1986, 83(5):1408-1412.
2. Taya Y, Devos R, Tavernier J, Cheroutre H, Engler G, Fiers W: Cloning and structure of the human immune interferon-gamma chromosomal gene. The EMBO journal 1982, 1(8):953-958.
3. Genomes Project C, Abecasis GR, Altshuler D, Auton A, Brooks LD, Durbin RM, Gibbs RA, Hurles ME, McVean GA: A map of human genome variation from population-scale sequencing. Nature 2010, 467(7319):1061-1073.
4. Ossowski S, Schneeberger K, Clark RM, Lanz C, Warthmann N, Weigel D: Sequencing of natural strains of Arabidopsis thaliana with short reads. Genome research 2008, 18(12):2024-2033.
5. Genome KCoS: Genome 10K: a proposal to obtain whole-genome sequence for 10,000 vertebrate species. The Journal of heredity 2009, 100(6):659-674.
6. Maxam AM, Gilbert W: A new method for sequencing DNA. Proceedings of the National Academy of Sciences of the United States of America 1977, 74(2):560-564.
7. Sanger F, Nicklen S, Coulson AR: DNA sequencing with chain-terminating inhibitors. Proceedings of the National Academy of Sciences of the United States of America 1977, 74(12):5463-5467.
8. NobelPrize: The nobel prize in chemistry. 1980.
9. Lander ES, Linton LM, Birren B, Nusbaum C, Zody MC, Baldwin J, Devon K, Dewar K, Doyle M, FitzHugh W, Funke R, Gage D, Harris K, Heaford A, Howland J, Kann L, Lehoczky J, LeVine R, McEwan P, McKernan K, Meldrim J, Mesirov JP, Miranda C, Morris W, Naylor J, Raymond C, Rosetti M, Santos R, Sheridan A, Sougnez C, Stange-Thomann N, Stojanovic N, Subramanian A, Wyman D, Rogers J, Sulston J, Ainscough R, Beck S, Bentley D, Burton J, Clee C, Carter N, Coulson A, Deadman R, Deloukas P, Dunham A, Dunham I, Durbin R, French L, Grafham D, Gregory S, Hubbard T, Humphray S, Hunt A, Jones M, Lloyd C, McMurray A, Matthews L, Mercer S, Milne S, Mullikin JC, Mungall A, Plumb R, Ross M, Shownkeen R, Sims S, Waterston RH, Wilson RK, Hillier LW, McPherson JD, Marra MA, Mardis ER, Fulton LA, Chinwalla AT, Pepin KH, Gish WR, Chissoe SL, Wendl MC, Delehaunty KD, Miner TL, Delehaunty A, Kramer JB, Cook LL, Fulton RS, Johnson DL, Minx PJ, Clifton SW, Hawkins T, Branscomb E, Predki P, Richardson P, Wenning S, Slezak T, Doggett N, Cheng JF, Olsen A, Lucas S, Elkin C, Uberbacher E, Frazier M, Gibbs RA, Muzny DM, Scherer SE, Bouck JB, Sodergren EJ, Worley KC, Rives CM, Gorrell JH, Metzker ML, Naylor SL, Kucherlapati RS, Nelson DL, Weinstock GM, Sakaki Y, Fujiyama A, Hattori M, Yada T, Toyoda A, Itoh T, Kawagoe C, Watanabe H, Totoki Y, Taylor T, Weissenbach J, Heilig R, Saurin W, Artiguenave F, Brottier P, Bruls T, Pelletier E, Robert C, Wincker P, Smith DR, Doucette-Stamm L, Rubenfield M, Weinstock K, Lee HM, Dubois J, Rosenthal A, Platzer M, Nyakatura G, Taudien S, Rump A, Yang H, Yu J, Wang J, Huang G, Gu J, Hood L, Rowen L, Madan A, Qin S, Davis RW, Federspiel NA,

Abola AP, Proctor MJ, Myers RM, Schmutz J, Dickson M, Grimwood J, Cox DR, Olson MV, Kaul R, Raymond C, Shimizu N, Kawasaki K, Minoshima S, Evans GA, Athanasiou M, Schultz R, Roe BA, Chen F, Pan H, Ramser J, Lehrach H, Reinhardt R, McCombie WR, de la Bastide M, Dedhia N, Blocker H, Hornischer K, Nordsiek G, Agarwala R, Aravind L, Bailey JA, Bateman A, Batzoglou S, Birney E, Bork P, Brown DG, Burge CB, Cerutti L, Chen HC, Church D, Clamp M, Copley RR, Doerks T, Eddy SR, Eichler EE, Furey TS, Galagan J, Gilbert JG, Harmon C, Hayashizaki Y, Haussler D, Hermjakob H, Hokamp K, Jang W, Johnson LS, Jones TA, Kasif S, Kaspryzk A, Kennedy S, Kent WJ, Kitts P, Koonin EV, Korf I, Kulp D, Lancet D, Lowe TM, McLysaght A, Mikkelsen T, Moran JV, Mulder N, Pollara VJ, Ponting CP, Schuler G, Schultz J, Slater G, Smit AF, Stupka E, Szustakowski J, Thierry-Mieg D, Thierry-Mieg J, Wagner L, Wallis J, Wheeler R, Williams A, Wolf YI, Wolfe KH, Yang SP, Yeh RF, Collins F, Guyer MS, Peterson J, Felsenfeld A, Wetterstrand KA, Patrinos A, Morgan MJ, de Jong P, Catanese JJ, Osoegawa K, Shizuya H, Choi S, Chen YJ, International Human Genome Sequencing C: Initial sequencing and analysis of the human genome. Nature 2001, 409(6822):860-921.

10. Venter JC, Adams MD, Myers EW, Li PW, Mural RJ, Sutton GG, Smith HO, Yandell M, Evans CA, Holt RA, Gocayne JD, Amanatides P, Ballew RM, Huson DH, Wortman JR, Zhang Q, Kodira CD, Zheng XH, Chen L, Skupski M, Subramanian G, Thomas PD, Zhang J, Gabor Miklos GL, Nelson C, Broder S, Clark AG, Nadeau J, McKusick VA, Zinder N, Levine AJ, Roberts RJ, Simon M, Slayman C, Hunkapiller M, Bolanos R, Delcher A, Dew I, Fasulo D, Flanigan M, Florea L, Halpern A, Hannenhalli S, Kravitz S, Levy S, Mobarry C, Reinert K, Remington K, Abu-Threideh J, Beasley E, Biddick K, Bonazzi V, Brandon R, Cargill M, Chandramouliswaran I, Charlab R, Chaturvedi K, Deng Z, Di Francesco V, Dunn P, Eilbeck K, Evangelista C, Gabrielian AE, Gan W, Ge W, Gong F, Gu Z, Guan P, Heiman TJ, Higgins ME, Ji RR, Ke Z, Ketchum KA, Lai Z, Lei Y, Li Z, Li J, Liang Y, Lin X, Lu F, Merkulov GV, Milshina N, Moore HM, Naik AK, Narayan VA, Neelam B, Nusskern D, Rusch DB, Salzberg S, Shao W, Shue B, Sun J, Wang Z, Wang A, Wang X, Wang J, Wei M, Wides R, Xiao C, Yan C, Yao A, Ye J, Zhan M, Zhang W, Zhang H, Zhao Q, Zheng L, Zhong F, Zhong W, Zhu S, Zhao S, Gilbert D, Baumhueter S, Spier G, Carter C, Cravchik A, Woodage T, Ali F, An H, Awe A, Baldwin D, Baden H, Barnstead M, Barrow I, Beeson K, Busam D, Carver A, Center A, Cheng ML, Curry L, Danaher S, Davenport L, Desilets R, Dietz S, Dodson K, Doup L, Ferriera S, Garg N, Gluecksmann A, Hart B, Haynes J, Haynes C, Heiner C, Hladun S, Hostin D, Houck J, Howland T, Ibegwam C, Johnson J, Kalush F, Kline L, Koduru S, Love A, Mann F, May D, McCawley S, McIntosh T, McMullen I, Moy M, Moy L, Murphy B, Nelson K, Pfannkoch C, Pratts E, Puri V, Qureshi H, Reardon M, Rodriguez R, Rogers YH, Romblad D, Ruhfel B, Scott R, Sitter C, Smallwood M, Stewart E, Strong R, Suh E, Thomas R, Tint NN, Tse S, Vech C, Wang G, Wetter J, Williams S, Williams M, Windsor S, Winn-Deen E, Wolfe K, Zaveri J, Zaveri K, Abril JF, Guigo R, Campbell MJ, Sjolander KV, Karlak B, Kejariwal A, Mi H, Lazareva B, Hatton T, Narechania A, Diemer K, Muruganujan A, Guo N, Sato S, Bafna V, Istrail S, Lippert R, Schwartz R, Walenz B, Yooseph S, Allen D, Basu A, Baxendale J, Blick L, Caminha M, Carnes-Stine J, Caulk P, Chiang YH, Coyne M,

Dahlke C, Mays A, Dombroski M, Donnelly M, Ely D, Esparham S, Fosler C, Gire H, Glanowski S, Glasser K, Glodek A, Gorokhov M, Graham K, Gropman B, Harris M, Heil J, Henderson S, Hoover J, Jennings D, Jordan C, Jordan J, Kasha J, Kagan L, Kraft C, Levitsky A, Lewis M, Liu X, Lopez J, Ma D, Majoros W, McDaniel J, Murphy S, Newman M, Nguyen T, Nguyen N, Nodell M, Pan S, Peck J, Peterson M, Rowe W, Sanders R, Scott J, Simpson M, Smith T, Sprague A, Stockwell T, Turner R, Venter E, Wang M, Wen M, Wu D, Wu M, Xia A, Zandieh A, Zhu X: The sequence of the human genome. Science 2001, 291(5507):1304-1351.

11. Zerbino DR, Birney E: Velvet: algorithms for de novo short read assembly using de Bruijn graphs. Genome research 2008, 18(5):821-829.
12. Margulies M, Egholm M, Altman WE, Attiya S, Bader JS, Bemben LA, Berka J, Braverman MS, Chen YJ, Chen Z, Dewell SB, Du L, Fierro JM, Gomes XV, Godwin BC, He W, Helgesen S, Ho CH, Irzyk GP, Jando SC, Alenquer ML, Jarvie TP, Jirage KB, Kim JB, Knight JR, Lanza JR, Leamon JH, Lefkowitz SM, Lei M, Li J, Lohman KL, Lu H, Makhijani VB, McDade KE, McKenna MP, Myers EW, Nickerson E, Nobile JR, Plant R, Puc BP, Ronan MT, Roth GT, Sarkis GJ, Simons JF, Simpson JW, Srinivasan M, Tartaro KR, Tomasz A, Vogt KA, Volkmer GA, Wang SH, Wang Y, Weiner MP, Yu P, Begley RF, Rothberg JM: Genome sequencing in microfabricated high-density picolitre reactors. Nature 2005, 437(7057):376-380.
13. Agah A, Aghajan M, Mashayekhi F, Amini S, Davis RW, Plummer JD, Ronaghi M, Griffin PB: A multi-enzyme model for Pyrosequencing. Nucleic acids research 2004, 32(21):e166.
14. Velasco R, Zharkikh A, Affourtit J, Dhingra A, Cestaro A, Kalyanaraman A, Fontana P, Bhatnagar SK, Troggio M, Pruss D, Salvi S, Pindo M, Baldi P, Castelletti S, Cavaiuolo M, Coppola G, Costa F, Cova V, Dal Ri A, Goremykin V, Komjanc M, Longhi S, Magnago P, Malacarne G, Malnoy M, Micheletti D, Moretto M, Perazzolli M, Si-Ammour A, Vezzulli S, Zini E, Eldredge G, Fitzgerald LM, Gutin N, Lanchbury J, Macalma T, Mitchell JT, Reid J, Wardell B, Kodira C, Chen Z, Desany B, Niazi F, Palmer M, Koepke T, Jiwan D, Schaeffer S, Krishnan V, Wu C, Chu VT, King ST, Vick J, Tao Q, Mraz A, Stormo A, Stormo K, Bogden R, Ederle D, Stella A, Vecchietti A, Kater MM, Masiero S, Lasserre P, Lespinasse Y, Allan AC, Bus V, Chagne D, Crowhurst RN, Gleave AP, Lavezzo E, Fawcett JA, Proost S, Rouze P, Sterck L, Toppo S, Lazzari B, Hellens RP, Durel CE, Gutin A, Bumgarner RE, Gardiner SE, Skolnick M, Egholm M, Van de Peer Y, Salamini F, Viola R: The genome of the domesticated apple (Malus x domestica Borkh.). Nature genetics 2010, 42(10):833-839.
15. Belknap WR, Wang Y, Huo N, Wu J, Rockhold DR, Gu YQ, Stover E: Characterizing the citrus cultivar Carrizo genome through 454 shotgun sequencing. Genome / National Research Council Canada = Genome / Conseil national de recherches Canada 2011, 54(12):1005-1015.
16. Velasco R, Zharkikh A, Troggio M, Cartwright DA, Cestaro A, Pruss D, Pindo M, Fitzgerald LM, Vezzulli S, Reid J, Malacarne G, Iliev D, Coppola G, Wardell B, Micheletti D, Macalma T, Facci M, Mitchell JT, Perazzolli M, Eldredge G, Gatto P, Oyzerski R, Moretto M, Gutin N, Stefanini M, Chen Y, Segala C, Davenport C, Dematte L, Mraz A, Battilana J, Stormo K, Costa F, Tao Q, Si-Ammour A, Harkins

T, Lackey A, Perbost C, Taillon B, Stella A, Solovyev V, Fawcett JA, Sterck L, Vandepoele K, Grando SM, Toppo S, Moser C, Lanchbury J, Bogden R, Skolnick M, Sgaramella V, Bhatnagar SK, Fontana P, Gutin A, Van de Peer Y, Salamini F, Viola R: A high quality draft consensus sequence of the genome of a heterozygous grapevine variety. PloS one 2007, 2(12):e1326.

17. Green RE, Krause J, Briggs AW, Maricic T, Stenzel U, Kircher M, Patterson N, Li H, Zhai W, Fritz MH, Hansen NF, Durand EY, Malaspinas AS, Jensen JD, Marques-Bonet T, Alkan C, Prufer K, Meyer M, Burbano HA, Good JM, Schultz R, Aximu-Petri A, Butthof A, Hober B, Hoffner B, Siegemund M, Weihmann A, Nusbaum C, Lander ES, Russ C, Novod N, Affourtit J, Egholm M, Verna C, Rudan P, Brajkovic D, Kucan Z, Gusic I, Doronichev VB, Golovanova LV, Lalueza-Fox C, de la Rasilla M, Fortea J, Rosas A, Schmitz RW, Johnson PL, Eichler EE, Falush D, Birney E, Mullikin JC, Slatkin M, Nielsen R, Kelso J, Lachmann M, Reich D, Paabo S: A draft sequence of the Neandertal genome. Science 2010, 328(5979):710-722.
18. Thomas Wicker ES, Andreas Graner, Timothy Close, Beat Keller, and Nils Stein: 454 sequencing put to the test using the complex genome of barley. BMC Genomics 2006, 7:275.
19. E. Novaes DD, W. Farmerie, G. Pappas, D. Grattapaglia, R. Sederoff, and M. Kirst: Highthroughput gene and snp discovery in eucalyptus grandis, an uncharacterized genome. BMC Genomics 2008, 9:312.
20. Barakat A, DiLoreto DS, Zhang Y, Smith C, Baier K, Powell WA, Wheeler N, Sederoff R, Carlson JE: Comparison of the transcriptomes of American chestnut (Castanea dentata) and Chinese chestnut (Castanea mollissima) in response to the chestnut blight infection. BMC plant biology 2009, 9:51.
21. Alagna F, D'Agostino N, Torchia L, Servili M, Rao R, Pietrella M, Giuliano G, Chiusano ML, Baldoni L, Perrotta G: Comparative 454 pyrosequencing of transcripts from two olive genotypes during fruit development. BMC Genomics 2009, 10:399.
22. Riggins CW, Peng Y, Stewart CN, Jr., Tranel PJ: Characterization of de novo transcriptome for waterhemp (Amaranthus tuberculatus) using GS-FLX 454 pyrosequencing and its application for studies of herbicide target-site genes. Pest management science 2010, 66(10):1042-1052.
23. Franssen SU, Shrestha RP, Brautigam A, Bornberg-Bauer E, Weber AP: Comprehensive transcriptome analysis of the highly complex Pisum sativum genome using next generation sequencing. BMC Genomics 2011, 12:227.
24. Angeloni F, Wagemaker CA, Jetten MS, Op den Camp HJ, Janssen-Megens EM, Francoijs KJ, Stunnenberg HG, Ouborg NJ: De novo transcriptome characterization and development of genomic tools for Scabiosa columbaria L. using next-generation sequencing techniques. Molecular ecology resources 2011, 11(4):662-674.
25. Swarbreck SM, Lindquist EA, Ackerly DD, Andersen GL: Analysis of leaf and root transcriptomes of soil-grown Avena barbata plants. Plant & cell physiology 2011, 52(2):317-332.
26. History of Solexa Sequencing [Online]. Available: http://www.illumina.com/technology/next-generation-sequencing/solexa-technology.html.

27. Turcatti G, Romieu A, Fedurco M, Tairi AP: A new class of cleavable fluorescent nucleotides: synthesis and optimization as reversible terminators for DNA sequencing by synthesis. Nucleic acids research 2008, 36(4):e25.
28. Fedurco M, Romieu A, Williams S, Lawrence I, Turcatti G: BTA, a novel reagent for DNA attachment on glass and efficient generation of solid-phase amplified DNA colonies. Nucleic acids research 2006, 34(3):e22.
29. Bentley DR, Balasubramanian S, Swerdlow HP, Smith GP, Milton J, Brown CG, Hall KP, Evers DJ, Barnes CL, Bignell HR, Boutell JM, Bryant J, Carter RJ, Keira Cheetham R, Cox AJ, Ellis DJ, Flatbush MR, Gormley NA, Humphray SJ, Irving LJ, Karbelashvili MS, Kirk SM, Li H, Liu X, Maisinger KS, Murray LJ, Obradovic B, Ost T, Parkinson ML, Pratt MR, Rasolonjatovo IM, Reed MT, Rigatti R, Rodighiero C, Ross MT, Sabot A, Sankar SV, Scally A, Schroth GP, Smith ME, Smith VP, Spiridou A, Torrance PE, Tzonev SS, Vermaas EH, Walter K, Wu X, Zhang L, Alam MD, Anastasi C, Aniebo IC, Bailey DM, Bancarz IR, Banerjee S, Barbour SG, Baybayan PA, Benoit VA, Benson KF, Bevis C, Black PJ, Boodhun A, Brennan JS, Bridgham JA, Brown RC, Brown AA, Buermann DH, Bundu AA, Burrows JC, Carter NP, Castillo N, Chiara ECM, Chang S, Neil Cooley R, Crake NR, Dada OO, Diakoumakos KD, Dominguez-Fernandez B, Earnshaw DJ, Egbujor UC, Elmore DW, Etchin SS, Ewan MR, Fedurco M, Fraser LJ, Fuentes Fajardo KV, Scott Furey W, George D, Gietzen KJ, Goddard CP, Golda GS, Granieri PA, Green DE, Gustafson DL, Hansen NF, Harnish K, Haudenschild CD, Heyer NI, Hims MM, Ho JT, Horgan AM, Hoschler K, Hurwitz S, Ivanov DV, Johnson MQ, James T, Huw Jones TA, Kang GD, Kerelska TH, Kersey AD, Khrebtukova I, Kindwall AP, Kingsbury Z, Kokko-Gonzales PI, Kumar A, Laurent MA, Lawley CT, Lee SE, Lee X, Liao AK, Loch JA, Lok M, Luo S, Mammen RM, Martin JW, McCauley PG, McNitt P, Mehta P, Moon KW, Mullens JW, Newington T, Ning Z, Ling Ng B, Novo SM, O'Neill MJ, Osborne MA, Osnowski A, Ostadan O, Paraschos LL, Pickering L, Pike AC, Pike AC, Chris Pinkard D, Pliskin DP, Podhasky J, Quijano VJ, Raczy C, Rae VH, Rawlings SR, Chiva Rodriguez A, Roe PM, Rogers J, Rogert Bacigalupo MC, Romanov N, Romieu A, Roth RK, Rourke NJ, Ruediger ST, Rusman E, Sanches-Kuiper RM, Schenker MR, Seoane JM, Shaw RJ, Shiver MK, Short SW, Sizto NL, Sluis JP, Smith MA, Ernest Sohna Sohna J, Spence EJ, Stevens K, Sutton N, Szajkowski L, Tregidgo CL, Turcatti G, Vandevondele S, Verhovsky Y, Virk SM, Wakelin S, Walcott GC, Wang J, Worsley GJ, Yan J, Yau L, Zuerlein M, Rogers J, Mullikin JC, Hurles ME, McCooke NJ, West JS, Oaks FL, Lundberg PL, Klenerman D, Durbin R, Smith AJ: Accurate whole human genome sequencing using reversible terminator chemistry. Nature 2008, 456(7218):53-59.
30. Adessi C, Matton G, Ayala G, Turcatti G, Mermod JJ, Mayer P, Kawashima E: Solid phase DNA amplification: characterisation of primer attachment and amplification mechanisms. Nucleic acids research 2000, 28(20):E87.
31. Strickler SR, Bombarely A, Mueller LA: Designing a transcriptome next-generation sequencing project for a nonmodel plant species. Am J Bot 2012, 99(2):257-266.
32. Caskey AEaCT: Closure strategies for random dna sequencing. Methods Comp Methods Enzymol 1991, 3:41-47.

33. Siegel AF, van den Engh G, Hood L, Trask B, Roach JC: Modeling the feasibility of whole genome shotgun sequencing using a pairwise end strategy. Genomics 2000, 68(3):237-246.
34. Mardis ER: Next-generation sequencing platforms. Annual review of analytical chemistry 2013, 6:287-303.
35. Davies JSCaE: Identifying adaptor contamination when mining dna sequence data. Biotechniques 2004, 37(2):194-198.
36. J. Falgueras AJL, N. Fernández-Pozo, F.R. Cantón, G. Pérez-Trabado, and M.G. Claros: Seqtrim: a high-throughput pipeline for pre-processing any type of sequence read. BMC Bioinformatics 2010, 11(38).
37. Li S, Chou HH: LUCY2: an interactive DNA sequence quality trimming and vector removal tool. Bioinformatics 2004, 20(16):2865-2866.
38. Scheetz TE, Trivedi N, Roberts CA, Kucaba T, Berger B, Robinson NL, Birkett CL, Gavin AJ, O'Leary B, Braun TA, Bonaldo MF, Robinson JP, Sheffield VC, Soares MB, Casavant TL: ESTprep: preprocessing cDNA sequence reads. Bioinformatics 2003, 19(11):1318-1324.
39. Schmieder R, Lim YW, Rohwer F, Edwards R: TagCleaner: Identification and removal of tag sequences from genomic and metagenomic datasets. BMC Bioinformatics 2010, 11:341.
40. Goecks J, Nekrutenko A, Taylor J, Galaxy T: Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome biology 2010, 11(8):R86.
41. Epstein SS: Uncultivated Microorganisms. Microbiology Monographs. Springer 2009.
42. Durfee T, Nelson R, Baldwin S, Plunkett G, 3rd, Burland V, Mau B, Petrosino JF, Qin X, Muzny DM, Ayele M, Gibbs RA, Csorgo B, Posfai G, Weinstock GM, Blattner FR: The complete genome sequence of Escherichia coli DH10B: insights into the biology of a laboratory workhorse. Journal of bacteriology 2008, 190(7):2597-2606.
43. Ishida Y, Saito H, Ohta S, Hiei Y, Komari T, Kumashiro T: High efficiency transformation of maize (Zea mays L.) mediated by Agrobacterium tumefaciens. Nature biotechnology 1996, 14(6):745-750.
44. Pennisi E: Keeping genome databases clean and up to date. Science 1999, 286(5439):447-450.
45. Schmieder R, Edwards R: Fast identification and removal of sequence contamination from genomic and metagenomic datasets. PloS one 2011, 6(3):e17288.
46. Fernández DDG: Plataforma de supercomputación para bioinformática. 264 Páginas. Departamento de Biología Molecular y Bioquímica Facultad de Ciencias. Plataforma Andaluza de Bioinformática, Edificio de Bioinnovación, Universidad de Málaga, Málaga; 2015.
47. Green ED: Strategies for the systematic sequencing of complex genomes. Nature reviews Genetics 2001, 2(8):573-583.
48. Pop M: Genome assembly reborn: recent computational challenges. Briefings in bioinformatics 2009, 10(4):354-366.
49. Pop M, Salzberg SL: Bioinformatics challenges of new sequencing technology. Trends in genetics : TIG 2008, 24(3):142-149.

50. Warren RL, Sutton GG, Jones SJ, Holt RA: Assembling millions of short DNA sequences using SSAKE. Bioinformatics 2007, 23(4):500-501.
51. Dohm JC, Lottaz C, Borodina T, Himmelbauer H: SHARCGS, a fast and highly accurate short-read assembly algorithm for de novo genomic sequencing. Genome research 2007, 17(11):1697-1706.
52. Jeck WR, Reinhardt JA, Baltrus DA, Hickenbotham MT, Magrini V, Mardis ER, Dangl JL, Jones CD: Extending assembly of short DNA sequences to handle error. Bioinformatics 2007, 23(21):2942-2944.
53. Reinhardt JA, Baltrus DA, Nishimura MT, Jeck WR, Jones CD, Dangl JL: De novo assembly using low-coverage short read sequence data from the rice pathogen Pseudomonas syringae pv. oryzae. Genome research 2009, 19(2):294-305.
54. Myers EW, Sutton GG, Delcher AL, Dew IM, Fasulo DP, Flanigan MJ, Kravitz SA, Mobarry CM, Reinert KH, Remington KA, Anson EL, Bolanos RA, Chou HH, Jordan CM, Halpern AL, Lonardi S, Beasley EM, Brandon RC, Chen L, Dunn PJ, Lai Z, Liang Y, Nusskern DR, Zhan M, Zhang Q, Zheng X, Rubin GM, Adams MD, Venter JC: A whole-genome assembly of Drosophila. Science 2000, 287(5461):2196-2204.
55. Goldberg SM, Johnson J, Busam D, Feldblyum T, Ferriera S, Friedman R, Halpern A, Khouri H, Kravitz SA, Lauro FM, Li K, Rogers YH, Strausberg R, Sutton G, Tallon L, Thomas T, Venter E, Frazier M, Venter JC: A Sanger/pyrosequencing hybrid approach for the generation of high-quality draft assemblies of marine microbial genomes. Proceedings of the National Academy of Sciences of the United States of America 2006, 103(30):11240-11245.
56. Batzoglou S, Jaffe DB, Stanley K, Butler J, Gnerre S, Mauceli E, Berger B, Mesirov JP, Lander ES: ARACHNE: a whole-genome shotgun assembler. Genome research 2002, 12(1):177-189.
57. Huang X, Wang J, Aluru S, Yang SP, Hillier L: PCAP: a whole-genome assembly program. Genome research 2003, 13(9):2164-2170.
58. Batzoglou S: Algorithmic challenges in mammalian genome sequence assembly, vol. 4; 2005.
59. Wang L, Jiang T: On the complexity of multiple sequence alignment. Journal of computational biology : a journal of computational molecular cell biology 1994, 1(4):337-348.
60. Hernandez D, Francois P, Farinelli L, Osteras M, Schrenzel J: De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer. Genome research 2008, 18(5):802-809.
61. Myers EW: Toward simplifying and accurately formulating fragment assembly. Journal of computational biology : a journal of computational molecular cell biology 1995, 2(2):275-290.
62. Hossain MS, Azimi N, Skiena S: Crystallizing short-read assemblies around seeds. BMC Bioinformatics 2009, 10 Suppl 1:S16.
63. Miller JR, Koren S, Sutton G: Assembly algorithms for next-generation sequencing data. Genomics 2010, 95(6):315-327.
64. Chikhi R, Medvedev P: Informed and automated k-mer size selection for genome assembly. Bioinformatics 2014, 30(1):31-37.

65. Pevzner PA, Tang H, Waterman MS: An Eulerian path approach to DNA fragment assembly. Proceedings of the National Academy of Sciences of the United States of America 2001, 98(17):9748-9753.
66. Li R, Yu C, Li Y, Lam TW, Yiu SM, Kristiansen K, Wang J: SOAP2: an improved ultrafast tool for short read alignment. Bioinformatics 2009, 25(15):1966-1967.
67. Simpson JT, Wong K, Jackman SD, Schein JE, Jones SJ, Birol I: ABySS: a parallel assembler for short read sequence data. Genome research 2009, 19(6):1117-1123.
68. Boisvert S, Laviolette F, Corbeil J: Ray: simultaneous assembly of reads from a mix of high-throughput sequencing technologies. Journal of computational biology : a journal of computational molecular cell biology 2010, 17(11):1519-1533.
69. Jain M: A next-generation approach to the characterization of a non-model plant transcriptome. CURRENT SCIENCE 2011, 101(11):1435–1439.
70. Cheung F, Haas BJ, Goldberg SM, May GD, Xiao Y, Town CD: Sequencing Medicago truncatula expressed sequenced tags using 454 Life Sciences technology. BMC Genomics 2006, 7:272.
71. Venter JC, Levy S, Stockwell T, Remington K, Halpern A: Massive parallelism, randomness and genomic advances. Nature genetics 2003, 33 Suppl:219-227.
72. Camacho C, Coulouris G, Avagyan V, Ma N, Papadopoulos J, Bealer K, Madden TL: BLAST+: architecture and applications. BMC Bioinformatics 2009, 10:421.
73. Benson DA, Boguski MS, Lipman DJ, Ostell J: GenBank. Nucleic acids research 1997, 25(1):1-6.
74. Apweiler R, Bairoch A, Wu CH, Barker WC, Boeckmann B, Ferro S, Gasteiger E, Huang H, Lopez R, Magrane M, Martin MJ, Natale DA, O'Donovan C, Redaschi N, Yeh LS: UniProt: the Universal Protein knowledgebase. Nucleic acids research 2004, 32(Database issue):D115-119.
75. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G: Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nature genetics 2000, 25(1):25-29.
76. Kanehisa M, Goto S, Sato Y, Furumichi M, Tanabe M: KEGG for integration and interpretation of large-scale molecular data sets. Nucleic acids research 2012, 40(Database issue):D109-114.
77. Hunter S, Jones P, Mitchell A, Apweiler R, Attwood TK, Bateman A, Bernard T, Binns D, Bork P, Burge S, de Castro E, Coggill P, Corbett M, Das U, Daugherty L, Duquenne L, Finn RD, Fraser M, Gough J, Haft D, Hulo N, Kahn D, Kelly E, Letunic I, Lonsdale D, Lopez R, Madera M, Maslen J, McAnulla C, McDowall J, McMenamin C, Mi H, Mutowo-Muellenet P, Mulder N, Natale D, Orengo C, Pesseat S, Punta M, Quinn AF, Rivoire C, Sangrador-Vegas A, Selengut JD, Sigrist CJ, Scheremetjew M, Tate J, Thimmajanarthanan M, Thomas PD, Wu CH, Yeats C, Yong SY: InterPro in 2011: new developments in the family and domain prediction database. Nucleic acids research 2012, 40(Database issue):D306-312.
78. Pruitt KD, Tatusova T, Maglott DR: NCBI Reference Sequence (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. Nucleic acids research 2005, 33(Database issue):D501-504.

79. Hubbard T, Barker D, Birney E, Cameron G, Chen Y, Clark L, Cox T, Cuff J, Curwen V, Down T, Durbin R, Eyras E, Gilbert J, Hammond M, Huminiecki L, Kasprzyk A, Lehvaslaiho H, Lijnzaad P, Melsopp C, Mongin E, Pettett R, Pocock M, Potter S, Rust A, Schmidt E, Searle S, Slater G, Smith J, Spooner W, Stabenau A, Stalker J, Stupka E, Ureta-Vidal A, Vastrik I, Clamp M: The Ensembl genome database project. Nucleic acids research 2002, 30(1):38-41.
80. Hamada H, Petrino MG, Kakunaga T: A novel repeated element with Z-DNA-forming potential is widely found in evolutionarily diverse eukaryotic genomes. Proceedings of the National Academy of Sciences of the United States of America 1982, 79(21):6465-6469.
81. Lee RC, Feinbaum RL, Ambros V: The C. elegans heterochronic gene lin-4 encodes small RNAs with antisense complementarity to lin-14. Cell 1993, 75(5):843-854.
82. Conesa A, Gotz S, Garcia-Gomez JM, Terol J, Talon M, Robles M: Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research. Bioinformatics 2005, 21(18):3674-3676.
83. Koski LB, Gray MW, Lang BF, Burger G: AutoFACT: an automatic functional annotation and classification tool. BMC Bioinformatics 2005, 6:151.
84. Munoz-Merida A, Viguera E, Claros MG, Trelles O, Perez-Pulido AJ: Sma3s: A Three-Step Modular Annotator for Large Sequence Datasets. DNA research : an international journal for rapid publication of reports on genes and genomes 2014.
85. Cantarel BL, Korf I, Robb SM, Parra G, Ross E, Moore B, Holt C, Sanchez Alvarado A, Yandell M: MAKER: an easy-to-use annotation pipeline designed for emerging model organism genomes. Genome research 2008, 18(1):188-196.
86. Holt C, Yandell M: MAKER2: an annotation pipeline and genome-database management tool for second-generation genome projects. BMC Bioinformatics 2011, 12:491.
87. Pozo NF: Análisis bioinformático del transcriptoma de pino, 317 páginas, . Departamento de Biología Molecular y Bioquímica Facultad de Ciencias. Plataforma Andaluza de Bioinformática, Edificio de Bioinnovación, Universidad de Málaga, Málaga; 2012.
88. Kanz C, Aldebert P, Althorpe N, Baker W, Baldwin A, Bates K, Browne P, van den Broek A, Castro M, Cochrane G, Duggan K, Eberhardt R, Faruque N, Gamble J, Diez FG, Harte N, Kulikova T, Lin Q, Lombard V, Lopez R, Mancuso R, McHale M, Nardone F, Silventoinen V, Sobhany S, Stoehr P, Tuli MA, Tzouvara K, Vaughan R, Wu D, Zhu W, Apweiler R: The EMBL Nucleotide Sequence Database. Nucleic acids research 2005, 33(Database issue):D29-33.
89. Tateno Y, Imanishi T, Miyazaki S, Fukami-Kobayashi K, Saitou N, Sugawara H, Gojobori T: DNA Data Bank of Japan (DDBJ) for genome scale research in life science. Nucleic acids research 2002, 30(1):27-30.
90. Lai D, Love DR: Automation of a primer design and evaluation pipeline for subsequent sequencing of the coding regions of all human Refseq genes. Bioinformation 2012, 8(8):365-368.
91. Kodama Y, Shumway M, Leinonen R, International Nucleotide Sequence Database C: The Sequence Read Archive: explosive growth of sequencing data. Nucleic acids research 2012, 40(Database issue):D54-56.

92. Leinonen R, Sugawara H, Shumway M, International Nucleotide Sequence Database C: The sequence read archive. Nucleic acids research 2011, 39(Database issue):D19-21.
93. Sprague J, Bayraktaroglu L, Clements D, Conlin T, Fashena D, Frazer K, Haendel M, Howe DG, Mani P, Ramachandran S, Schaper K, Segerdell E, Song P, Sprunger B, Taylor S, Van Slyke CE, Westerfield M: The Zebrafish Information Network: the zebrafish model organism database. Nucleic acids research 2006, 34(Database issue):D581-585.
94. Aerts J, Law A: An introduction to scripting in Ruby for biologists. BMC Bioinformatics 2009, 10:221.
95. FAO: Report of the Conference on Aquaculture in the Third Millenium, Bangkok, Tailandia, 20-25 February 2000. In.; 2001.
96. FAO: The State of World Fisheries and Aquaculture. Rome. In.; 2002.
97. Norum KR: Fish and human health. In. European Aquaculture Society Meeting, Barcelona (Spain)
2004: 49-50.
98. Simopoulos AP: Evolutionary aspects of diet, the omega-6/omega-3 ratio and genetic variation: nutritional implications for chronic diseases. Biomedicine & pharmacotherapy = Biomedecine & pharmacotherapie 2006, 60(9):502-507.
99. FAO: El estado mundial de la pesca y la acuicultura 2012. Roma. 209 pages. In.; 2012.
100. Divanach P: New species in Mediterranean, dream or reality. In: RTD needs in Mediterranean fish farming. Atenas, Grecia; 2003.
101. Howell BR: A re-appraisal of the potential of the sole, Solea solea (L.), for commercial cultivation. Aquaculture 1997, 155:355-365.
102. Rodríguez Martínez RB: Biología y Cultivo de Solea senegalensis Kaup 1858 en el Golfo de Cádiz. Sevilla: Sevilla; 1984.
103. Asociación Empresarial de Productores de Cultivos Marinos de España (APROMAR): La Acuicultura en España. 2014.
104. Goto N, Prins P, Nakao M, Bonnal R, Aerts J, Katayama T: BioRuby: bioinformatics software for the Ruby programming language. Bioinformatics 2010, 26(20):2617-2619.
105. Huang X, Madan A: CAP3: A DNA sequence assembly program. Genome research 1999, 9(9):868-877.
106. Liang F, Holt I, Pertea G, Karamycheva S, Salzberg SL, Quackenbush J: An optimized protocol for analysis of EST sequences. Nucleic acids research 2000, 28(18):3657-3665.
107. Zheng Y, Zhao L, Gao J, Fei Z: iAssembler: a package for de novo assembly of Roche-454/Sanger transcriptome sequences. BMC Bioinformatics 2011, 12:453.
108. Kumar S, Blaxter ML: Comparing de novo assemblers for 454 transcriptome data. BMC Genomics 2010, 11:571.
109. Chevreux B, Pfisterer T, Drescher B, Driesel AJ, Muller WE, Wetter T, Suhai S: Using the miraEST assembler for reliable and automated mRNA transcript assembly and SNP detection in sequenced ESTs. Genome research 2004, 14(6):1147-1159.
110. Chaisson M, Pevzner P, Tang H: Fragment assembly with short reads. Bioinformatics 2004, 20(13):2067-2074.

111. Chaisson MJ, Pevzner PA: Short read fragment assembly of bacterial genomes. Genome research 2008, 18(2):324-330.
112. Chaisson MJ, Brinza D, Pevzner PA: De novo fragment assembly with short mate-paired reads: Does the read length matter? Genome research 2009, 19(2):336-346.
113. Lander ES, Waterman MS: Genomic mapping by fingerprinting random clones: a mathematical analysis. Genomics 1988, 2(3):231-239.
114. Zerbino DR, McEwen GK, Margulies EH, Birney E: Pebble and rock band: heuristic resolution of repeats and scaffolding in the velvet short-read de novo assembler. PloS one 2009, 4(12):e8407.
115. Schulz MH, Zerbino DR, Vingron M, Birney E: Oases: robust de novo RNA-seq assembly across the dynamic range of expression levels. Bioinformatics 2012, 28(8):1086-1092.
116. Luo R, Liu B, Xie Y, Li Z, Huang W, Yuan J, He G, Chen Y, Pan Q, Liu Y, Tang J, Wu G, Zhang H, Shi Y, Liu Y, Yu C, Wang B, Lu Y, Han C, Cheung DW, Yiu SM, Peng S, Xiaoqian Z, Liu G, Liao X, Li Y, Yang H, Wang J, Lam TW, Wang J: SOAPdenovo2: an empirically improved memory-efficient short-read de novo assembler. GigaScience 2012, 1(1):18.
117. Forum MPI: MPI-2: Extensions to the Message-Passing Interface. 1997.
118. Vicedomini R, Vezzi F, Scalabrin S, Arvestad L, Policriti A: GAM-NGS: genomic assemblies merger for next generation sequencing. BMC Bioinformatics 2013, 14 Suppl 7:S6.
119. Langmead B, Trapnell C, Pop M, Salzberg SL: Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome biology 2009, 10(3):R25.
120. Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R, Genome Project Data Processing S: The Sequence Alignment/Map format and SAMtools. Bioinformatics 2009, 25(16):2078-2079.
121. Boetzer M, Henkel CV, Jansen HJ, Butler D, Pirovano W: Scaffolding pre-assembled contigs using SSPACE. Bioinformatics 2011, 27(4):578-579.
122. Li H, Durbin R: Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics 2009, 25(14):1754-1760.
123. A. Lara GP-T, D. Villalobos, S. Díaz-Moreno, F. Cantón, and M. G. Claros: A Web Tool to Discover Full-Length Sequences: Full-Lengther. Springer 2007:361–368.
124. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ: Basic local alignment search tool. Journal of molecular biology 1990, 215(3):403-410.
125. Aziz RK, Bartels D, Best AA, DeJongh M, Disz T, Edwards RA, Formsma K, Gerdes S, Glass EM, Kubal M, Meyer F, Olsen GJ, Olson R, Osterman AL, Overbeek RA, McNeil LK, Paarmann D, Paczian T, Parrello B, Pusch GD, Reich C, Stevens R, Vassieva O, Vonstein V, Wilke A, Zagnitko O: The RAST Server: rapid annotations using subsystems technology. BMC Genomics 2008, 9:75.
126. Lindner R, Friedel CC: A comprehensive evaluation of alignment algorithms in the context of RNA-seq. PloS one 2012, 7(12):e52403.
127. Cock PJ, Fields CJ, Goto N, Heuer ML, Rice PM: The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. Nucleic acids research 2010, 38(6):1767-1771.

128. Danecek P, Auton A, Abecasis G, Albers CA, Banks E, DePristo MA, Handsaker RE, Lunter G, Marth GT, Sherry ST, McVean G, Durbin R, Genomes Project Analysis G: The variant call format and VCFtools. Bioinformatics 2011, 27(15):2156-2158.
129. Li W, Godzik A: Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. Bioinformatics 2006, 22(13):1658-1659.
130. Kolpakov R, Bana G, Kucherov G: mreps: Efficient and flexible detection of tandem repeats in DNA. Nucleic acids research 2003, 31(13):3672-3678.
131. Marth GT: Boston College, http://bioinformatics.bc.edu/marthlab.
132. Milne I, Bayer M, Cardle L, Shaw P, Stephen G, Wright F, Marshall D: Tablet--next generation sequence assembly visualization. Bioinformatics 2010, 26(3):401-402.
133. Schanable JC, Lyons E: Comparative genomics with maize and other grasses: from genes to genomes! Maydica 2001, 56.
134. Kurtz S, Phillippy A, Delcher AL, Smoot M, Shumway M, Antonescu C, Salzberg SL: Versatile and open software for comparing large genomes. Genome biology 2004, 5(2):R12.
135. Fickett JW: Recognition of protein coding regions in DNA sequences. Nucleic acids research 1982, 10(17):5303-5318.
136. Min XJ, Butler G, Storms R, Tsang A: OrfPredictor: predicting protein-coding regions in EST-derived sequences. Nucleic acids research 2005, 33(Web Server issue):W677-680.
137. Epstein SS: Uncultivated Microorganisms. Microbiology Monographs Springer 2009.
138. Miller. BFaJD: Health implications of fungi in indoor environments - an overview. In Health Implications of Fungi in Indoor Environments. Elsevier 1993.
139. Chevreux B, Wetter, T., and Suhai: Genome sequence assembly using trace signals and additional sequence information. Comput Sci Biol 1999, German Conference on Bioinformatics GCB'99 GCB:45-56.
140. Carmona R, Zafra A, Seoane P, Castro AJ, Guerrero-Fernandez D, Castillo-Castillo T, Medina-Garcia A, Canovas FM, Aldana-Montes JF, Navas-Delgado I, Alche Jde D, Claros MG: ReprOlive: a database with linked data for the olive tree (Olea europaea L.) reproductive transcriptome. Frontiers in plant science 2015, 6:625.
141. Fernandez-Pozo N, Canales J, Guerrero-Fernandez D, Villalobos DP, Diaz-Moreno SM, Bautista R, Flores-Monterroso A, Guevara MA, Perdiguero P, Collada C, Cervera MT, Soto A, Ordas R, Canton FR, Avila C, Canovas FM, Claros MG: EuroPineDB: a high-coverage web database for maritime pine transcriptome. BMC Genomics 2011, 12:366.
142. Seoane P, Carmona R, Bautista R, Guerrero D, Claros GM: AutoFlow: an easy way to build workflows. International Work-Conference on Bioinformatics and Biomedical Engineering IWBBIO, Granada 2014:8.
143. Benzekri H, Armesto P, Cousin X, Rovira M, Crespo D, Merlo MA, Mazurais D, Bautista R, Guerrero-Fernandez D, Fernandez-Pozo N, Ponce M, Infante C, Zambonino JL, Nidelet S, Gut M, Rebordinos L, Planas JV, Begout ML, Claros MG, Manchado M: De novo assembly, characterization and functional annotation of Senegalese sole (Solea senegalensis) and common sole (Solea solea) transcriptomes: integration in a database and design of a microarray. BMC Genomics 2014, 15:952.

144. Soto-Jimenez LM, Estrada K, Sanchez-Flores A: GARM: genome assembly, reconciliation and merging pipeline. Current topics in medicinal chemistry 2014, 14(3):418-424.
145. Istrail S, Sutton GG, Florea L, Halpern AL, Mobarry CM, Lippert R, Walenz B, Shatkay H, Dew I, Miller JR, Flanigan MJ, Edwards NJ, Bolanos R, Fasulo D, Halldorsson BV, Hannenhalli S, Turner R, Yooseph S, Lu F, Nusskern DR, Shue BC, Zheng XH, Zhong F, Delcher AL, Huson DH, Kravitz SA, Mouchard L, Reinert K, Remington KA, Clark AG, Waterman MS, Eichler EE, Adams MD, Hunkapiller MW, Myers EW, Venter JC: Whole-genome shotgun assembly and comparison of human genome assemblies. Proceedings of the National Academy of Sciences of the United States of America 2004, 101(7):1916-1921.
146. Phillippy AM, Schatz MC, Pop M: Genome assembly forensics: finding the elusive mis-assembly. Genome biology 2008, 9(3):R55.
147. Schatz MC, Phillippy AM, Shneiderman B, Salzberg SL: Hawkeye: an interactive visual analytics tool for genome assemblies. Genome biology 2007, 8(3):R34.
148. Bonfield JK, Whitwham A: Gap5--editing the billion fragment sequence assembly. Bioinformatics 2010, 26(14):1699-1703.
149. Gordon D, Abajian C, Green P: Consed: a graphical tool for sequence finishing. Genome research 1998, 8(3):195-202.
150. H. B, D. G-F, R. B, M.G. C: Detecting and correcting mis-assembled reads in contigs. iwbbio Granada 2013.
151. Guerrero D, Bautista R, Villalobos DP, Canton FR, Claros MG: AlignMiner: a Web-based tool for detection of divergent regions in multiple sequence alignments of conserved sequences. Algorithms for molecular biology : AMB 2010, 5:24.
152. Falgueras J, Lara AJ, Fernandez-Pozo N, Canton FR, Perez-Trabado G, Claros MG: SeqTrim: a high-throughput pipeline for pre-processing any type of sequence read. BMC Bioinformatics 2010, 11:38.
153. Langmead B, Salzberg SL: Fast gapped-read alignment with Bowtie 2. Nature methods 2012, 9(4):357-359.
154. Richter DC, Schuster SC, Huson DH: OSLay: optimal syntenic layout of unfinished assemblies. Bioinformatics 2007, 23(13):1573-1579.
155. van Hijum SA, Zomer AL, Kuipers OP, Kok J: Projector 2: contig mapping for efficient gap-closure of prokaryotic genome sequence assemblies. Nucleic acids research 2005, 33(Web Server issue):W560-566.
156. Rissman AI, Mau B, Biehl BS, Darling AE, Glasner JD, Perna NT: Reordering contigs of draft genomes using the Mauve aligner. Bioinformatics 2009, 25(16):2071-2073.
157. Pertea G, Huang X, Liang F, Antonescu V, Sultana R, Karamycheva S, Lee Y, White J, Cheung F, Parvizi B, Tsai J, Quackenbush J: TIGR Gene Indices clustering tools (TGICL): a software system for fast clustering of large EST datasets. Bioinformatics 2003, 19(5):651-652.
158. Canales J, Bautista R, Label P, Gomez-Maldonado J, Lesur I, Fernandez-Pozo N, Rueda-Lopez M, Guerrero-Fernandez D, Castro-Rodriguez V, Benzekri H, Canas RA, Guevara MA, Rodrigues A, Seoane P, Teyssier C, Morel A, Ehrenmann F, Le Provost G, Lalanne C, Noirot C, Klopp C, Reymond I, Garcia-Gutierrez A, Trontin JF, Lelu-

Walter MA, Miguel C, Cervera MT, Canton FR, Plomion C, Harvengt L, Avila C, Gonzalo Claros M, Canovas FM: De novo assembly of maritime pine transcriptome: implications for forest breeding and biotechnology. Plant biotechnology journal 2014, 12(3):286-299.

159. Howe K, Clark MD, Torroja CF, Torrance J, Berthelot C, Muffato M, Collins JE, Humphray S, McLaren K, Matthews L, McLaren S, Sealy I, Caccamo M, Churcher C, Scott C, Barrett JC, Koch R, Rauch GJ, White S, Chow W, Kilian B, Quintais LT, Guerra-Assuncao JA, Zhou Y, Gu Y, Yen J, Vogel JH, Eyre T, Redmond S, Banerjee R, Chi J, Fu B, Langley E, Maguire SF, Laird GK, Lloyd D, Kenyon E, Donaldson S, Sehra H, Almeida-King J, Loveland J, Trevanion S, Jones M, Quail M, Willey D, Hunt A, Burton J, Sims S, McLay K, Plumb B, Davis J, Clee C, Oliver K, Clark R, Riddle C, Elliot D, Threadgold G, Harden G, Ware D, Begum S, Mortimore B, Kerry G, Heath P, Phillimore B, Tracey A, Corby N, Dunn M, Johnson C, Wood J, Clark S, Pelan S, Griffiths G, Smith M, Glithero R, Howden P, Barker N, Lloyd C, Stevens C, Harley J, Holt K, Panagiotidis G, Lovell J, Beasley H, Henderson C, Gordon D, Auger K, Wright D, Collins J, Raisen C, Dyer L, Leung K, Robertson L, Ambridge K, Leongamornlert D, McGuire S, Gilderthorp R, Griffiths C, Manthravadi D, Nichol S, Barker G, Whitehead S, Kay M, Brown J, Murnane C, Gray E, Humphries M, Sycamore N, Barker D, Saunders D, Wallis J, Babbage A, Hammond S, Mashreghi-Mohammadi M, Barr L, Martin S, Wray P, Ellington A, Matthews N, Ellwood M, Woodmansey R, Clark G, Cooper J, Tromans A, Grafham D, Skuce C, Pandian R, Andrews R, Harrison E, Kimberley A, Garnett J, Fosker N, Hall R, Garner P, Kelly D, Bird C, Palmer S, Gehring I, Berger A, Dooley CM, Ersan-Urun Z, Eser C, Geiger H, Geisler M, Karotki L, Kirn A, Konantz J, Konantz M, Oberlander M, Rudolph-Geiger S, Teucke M, Lanz C, Raddatz G, Osoegawa K, Zhu B, Rapp A, Widaa S, Langford C, Yang F, Schuster SC, Carter NP, Harrow J, Ning Z, Herrero J, Searle SM, Enright A, Geisler R, Plasterk RH, Lee C, Westerfield M, de Jong PJ, Zon LI, Postlethwait JH, Nusslein-Volhard C, Hubbard TJ, Roest Crollius H, Rogers J, Stemple DL: The zebrafish reference genome sequence and its relationship to the human genome. Nature 2013, 496(7446):498-503.

160. Chen S, Zhang G, Shao C, Huang Q, Liu G, Zhang P, Song W, An N, Chalopin D, Volff JN, Hong Y, Li Q, Sha Z, Zhou H, Xie M, Yu Q, Liu Y, Xiang H, Wang N, Wu K, Yang C, Zhou Q, Liao X, Yang L, Hu Q, Zhang J, Meng L, Jin L, Tian Y, Lian J, Yang J, Miao G, Liu S, Liang Z, Yan F, Li Y, Sun B, Zhang H, Zhang J, Zhu Y, Du M, Zhao Y, Schartl M, Tang Q, Wang J: Whole-genome sequence of a flatfish provides insights into ZW sex chromosome evolution and adaptation to a benthic lifestyle. Nature genetics 2014, 46(3):253-260.

161. Zhao QY, Wang Y, Kong YM, Luo D, Li X, Hao P: Optimizing de novo transcriptome assembly from short-read RNA-Seq data: a comparative study. BMC Bioinformatics 2011, 12 Suppl 14:S2.

162. Cahais V, Gayral P, Tsagkogeorga G, Melo-Ferreira J, Ballenghien M, Weinert L, Chiari Y, Belkhir K, Ranwez V, Galtier N: Reference-free transcriptome assembly in non-model animals from next-generation sequencing data. Molecular ecology resources 2012, 12(5):834-845.

163. Shen Y, Wan Z, Coarfa C, Drabek R, Chen L, Ostrowski EA, Liu Y, Weinstock GM, Wheeler DA, Gibbs RA, Yu F: A SNP discovery method to assess variant allele probability from next-generation resequencing data. Genome research 2010, 20(2):273-280.
164. Li H: A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. Bioinformatics 2011, 27(21):2987-2993.
165. Francis WR, Christianson LM, Kiko R, Powers ML, Shaner NC, SH DH: A comparison across non-model animals suggests an optimal sequencing depth for de novo transcriptome assembly. BMC Genomics 2013, 14:167.
166. Kettleborough RN, Busch-Nentwich EM, Harvey SA, Dooley CM, de Bruijn E, van Eeden F, Sealy I, White RJ, Herd C, Nijman IJ, Fenyes F, Mehroke S, Scahill C, Gibbons R, Wali N, Carruthers S, Hall A, Yen J, Cuppen E, Stemple DL: A systematic genome-wide analysis of zebrafish protein-coding gene function. Nature 2013, 496(7446):494-497.
167. Toth G, Gaspari Z, Jurka J: Microsatellites in different eukaryotic genomes: survey and analysis. Genome research 2000, 10(7):967-981.
168. Liu SR, Li WY, Long D, Hu CG, Zhang JZ: Development and characterization of genomic and expressed SSRs in citrus by genome-wide analysis. PloS one 2013, 8(10):e75149.
169. Brenner S, Elgar G, Sandford R, Macrae A, Venkatesh B, Aparicio S: Characterization of the pufferfish (Fugu) genome as a compact model vertebrate genome. Nature 1993, 366(6452):265-268.
170. Ferraresso S, Bonaldo A, Parma L, Cinotti S, Massi P, Bargelloni L, Gatta PP: Exploring the larval transcriptome of the common sole (Solea solea L.). BMC Genomics 2013, 14:315.
171. Vera M, Alvarez-Dios J, Millán A, Pardo B, Bouza C, Hermida M, Fernández C, de la Herran R, Molina-Luzon M, Martínez P: Validation of single nucleotide polymorphism (SNP) markers from an immune Expressed Sequence Tag (EST) turbot, Scophthalmus maximus, database. Aquaculture 2011, 313:31-41.
172. Hayes B, Laerdahl J, Lien S, Moen T, Berg P, Hindar K, Davidson W, Koop B, Adzhubei A, Hoyheim B: An extensive resource of single nucleotide polymorphism markers associated with Atlantic salmon (Salmo salar) expressed. Aquaculture 2007, 265:82-90.
173. Hubert S, Bussey J, Higgins B, Curtis B, Bowman S: Development of single nucleotide polymorphism markers for Atlantic cod (Gadus morhua) using expressed sequences. Aquaculture 2009, 296:7-14.
174. Che R, Sun Y, Wang R, Xu T: Transcriptomic analysis of endangered Chinese salamander: identification of immune, sex and reproduction-related genes and genetic markers. PloS one 2014, 9(1):e87940.
175. Cerda J, Manchado M: Advances in genomics for flatfish aquaculture. Genes & nutrition 2013, 8(1):5-17.
176. Spolaore P, Joannis-Cassan C, Duran E, Isambert A: Commercial applications of microalgae. Journal of bioscience and bioengineering 2006, 101(2):87-96.

177. Sili G, Torzillo G, Vonshak A: Ecology of Cyanobacteria II. Springer Netherlands 2012:677-705.
178. Bendif E, Probert I, Schroeder D, De Vegas C: On the description of Tisochrysis lutea gen. nov. sp. nov. and Isochrysis nuda sp. nov. in the Isochrysidales, and the transfer of Dicrateria to the Prymnesiales (Haptophyta). Springer 2013, 25:1763-1776.
179. Xie Y, Wu G, Tang J, Luo R, Patterson J, Liu S, Huang W, He G, Gu S, Li S, Zhou X, Lam TW, Li Y, Xu X, Wong GK, Wang J: SOAPdenovo-Trans: de novo transcriptome assembly with short RNA-Seq reads. Bioinformatics 2014, 30(12):1660-1666.
180. Houdan A, Billard C, Marie D, Not F, Sáez A, Young JR, Probert I: Holococcolithophore-heterococcolithophore (Haptophyta) life cycles: flow cytometric analysis of relative ploidy levels. Systematics and Biodiverity 2004, 1(4):453-465.
181. Carrier G, Garnier M, Le Cunff L, Bougaran G, Probert I, De Vargas C, Corre E, Cadoret JP, Saint-Jean B: Comparative transcriptome of wild type and selected strains of the microalgae Tisochrysis lutea provides insights into the genetic basis, lipid metabolism and the life cycle. PloS one 2014, 9(1):e86889.
182. Kroeker KJ, Kordas RL, Crim RN, Singh GG: Meta-analysis reveals negative yet variable effects of ocean acidification on marine organisms. Ecology letters 2010, 13(11):1419-1434.
183. Rodríguez-Moscoso E, Arnáiz R: Gametogenis and energy storage in a population of the grooved carpet- shell clam, Tapes decussatus (Linné,1787), in nothwest Spain Aquaculture 1998, 162:125-139.
184. Milan M, Coppe A, Reinhardt R, Cancela LM, Leite RB, Saavedra C, Ciofi C, Chelazzi G, Patarnello T, Bortoluzzi S, Bargelloni L: Transcriptome sequencing and microarray development for the Manila clam, Ruditapes philippinarum: genomic tools for environmental monitoring. BMC Genomics 2011, 12:234.
185. Aarestrup FM: Antimicrobial Resistance in Bacteria of Animal Origin ASM Press, Washington DC 2006, 12:7.
186. Zorilla I, Balebona MC, Moriñigo MA, Sarasquete C, Borrego JJ: Isolation and characterization of the causative agent of pasteurellosis, Photobacterium damselae ssp. piscicida, from sole, Solea senegalensis (Kaup). Fish Diseases 1999, 22(3):167-172.
187. Osorio CR, Rivas AJ, Balado M, Fuentes-Monteverde JC, Rodriguez J, Jimenez C, Lemos ML, Waldor MK: A Transmissible Plasmid-Borne Pathogenicity Island Confers Piscibactin Biosynthesis in the Fish Pathogen Photobacterium damselae subsp. piscicida. Applied and environmental microbiology 2015, 81(17):5867-5879.
188. Hunt M, Newbold C, Berriman M, Otto TD: A comprehensive evaluation of assembly scaffolding tools. Genome biology 2014, 15(3):R42.
189. Kisand V, Lettieri T: Genome sequencing of bacteria: sequencing, de novo assembly and rapid analysis using open source tools. BMC Genomics 2013, 14:211.
190. Donlin M: Using the Generic Genome Browser (GBrowse). Current Protocols in Bioinformatics 2009, 9(Unit 9.9).
191. Krzywinski M, Schein J, Birol I, Connors J, Gascoyne R, Horsman D, Jones SJ, Marra MA: Circos: an information aesthetic for comparative genomics. Genome research 2009, 19(9):1639-1645.

192. Overbeek R, Begley T, Butler RM, Choudhuri JV, Chuang HY, Cohoon M, de Crecy-Lagard V, Diaz N, Disz T, Edwards R, Fonstein M, Frank ED, Gerdes S, Glass EM, Goesmann A, Hanson A, Iwata-Reuyl D, Jensen R, Jamshidi N, Krause L, Kubal M, Larsen N, Linke B, McHardy AC, Meyer F, Neuweger H, Olsen G, Olson R, Osterman A, Portnoy V, Pusch GD, Rodionov DA, Ruckert C, Steiner J, Stevens R, Thiele I, Vassieva O, Ye Y, Zagnitko O, Vonstein V: The subsystems approach to genome annotation and its use in the project to annotate 1000 genomes. Nucleic acids research 2005, 33(17):5691-5702.

193. Parra G, Bradnam K, Korf I: CEGMA: a pipeline to accurately annotate core genes in eukaryotic genomes. Bioinformatics 2007, 23(9):1061-1067.

194. Kapustin Y, Souvorov A, Tatusova T, Lipman D: Splign: algorithms for computing spliced alignments with identification of paralogs. Biology direct 2008, 3:20.

195. Molina-Luzon MJ, Hermida M, Navajas-Perez R, Robles F, Navas JI, Ruiz-Rejon C, Bouza C, Martinez P, de la Herran R: First haploid genetic map based on microsatellite markers in Senegalese sole (Solea senegalensis, Kaup 1858). Marine biotechnology 2015, 17(1):8-22.

196. Molina-Luzon MJ, Lopez JR, Navajas-Perez R, Robles F, Ruiz-Rejon C, De La Herran R: Validation and comparison of microsatellite markers derived from Senegalese sole (Solea senegalensis, Kaup) genomic and expressed sequence tags libraries. Molecular ecology resources 2012, 12(5):956-966.

Parte VII
Apéndices

VII. Apéndices

Apendice A: Script para crear lecturas artificiales

```
#!/usr/bin/env perl
#Hicham Benzekri
#2010

#_____________________________________ import seq module of bioperl

use Bio::SeqIO;
#_____________________________________ Input files

my $seq_file   = $ARGV[0];
my $min_rd_length = $ARGV[1];
my $max_rd_length = $ARGV[2];
my $coverage  = $ARGV[42];

my $string_for_60;
my $number_qual;
my $x = 1;

# Creat folders and fasta files
my $folder="$min_rd_length.$max_rd_length.$coverage";
system ("mkdir ./reads/$folder");
open( my $file_fasta, ">./reads/$folder/reads.fasta" );
open( my $file_qual, ">./reads/$folder/reads.qual" );

my $seqio_obj = Bio::SeqIO->new(
      -file   => $seq_file,
      -format => "fasta"
);

#_____________________________________Parse fasta file sequences

while ( my $secuencia = $seqio_obj->next_seq ) {

      #_____________________________variables
      my $max    = 0;
      my $sumRds = 0;
      my $length;
      my $seq;
      my $read;

      $seq = $secuencia->seq;
      my $id=$secuencia->id;
      $length = length($seq);

      my $minimum = 10000;
      my $maximum = 0;

      #_____________________________ Print reads
      if ( $length <= $max_rd_length ) {
            print $file_fasta ">Read_$x\n";
            print $file_qual ">Read_$x\n";
```

```
52              $string_for_60 = $seq;
53              &string_lines_60;
54              $number_qual=length($seq);
55              &make_qual;
56              $x++;
57        }
58        else {
59          #_____________________________ Creat random position
60              my $lRd = int( rand( $max_rd_length - $min_rd_length ) +
61              $min_rd_length );
62              $read = substr( $seq, 0, $lRd );
63              $sumRds += (2*$lRd);
64               print $file_fasta ">Read_$x\n";
65               print $file_qual ">Read_$x\n";
66
67               $string_for_60 = $read;
68               &string_lines_60;
69               $number_qual=$lRd;
70               &make_qual;
71               $x++;
72
73               $read = substr( $seq, ($length-$lRd), $lRd );
74               print $file_fasta ">Read_$x\n";
75               print $file_qual ">Read_$x\n";
76               $string_for_60 = $read;
77               &string_lines_60;
78               $number_qual=$lRd;
79               &make_qual;
80               $x++;
81
82              #_______________________________Creat random position and print reads
83
84               while ( $max <= $coverage ) {
85                     my $pos = int( rand( $length - $max_rd_length ) );
86                     my $lRd = int( rand( $max_rd_length - $min_rd_length ) +
87                    $min_rd_length );
88                     $read = substr( $seq, $pos, $lRd );
89                    $sumRds += $lRd;
90                    if ( $pos > $maximum ) { $maximum = $pos }
91                    if ( $pos < $minimum ) { $minimum = $pos }
92                    $max = $sumRds / $length;
93                    print $file_fasta ">Read_$x\n";
94                    print $file_qual ">Read_$x\n";
95                    $string_for_60 = $read;
96                    &string_lines_60;
97                    $number_qual=$lRd;
98                    &make_qual;
99                    $x++;
100             }
101       }
102 }
103
104 # Creat qualities value of 60 for all nucleotides
105 sub make_qual {
106       for (my $i=1;$i<=$number_qual;$i++){
107             print $file_qual "60 ";
108             if (int($i/25)==($i/25)){
```

```
109                     print $file_qual "\n";
110               }
111         }
112         print $file_qual "\n";
113   }
114
115   #_____________________________ Fonction for making lines of 60 nucleotides
116   length
117   sub string_lines_60 {
118
119         my $l     = length($string_for_60);
120         my $after = 0;
121         while ( $l > 0 ) {
122               my $part = substr( $string_for_60, $after, 60 );
123               my $long_part = length($part);
124               print $file_fasta "$part\n";
125               $after = $after + $long_part;
126               $l     = $l - $long_part;
127         }
128   }
```

Apendice B: Script para calcular el tamaño de inserto en lecturas Illumina a partir de un fichero de mapeo (SAM)

```
#!/usr/bin/env ruby
# Hicham benzekri
# 05-02-2013
# Script para conseguir el tamano de inserto de lecturas pareadas a partir de un
# fichero de mapping de estas lecturas sobre una referencia (SAM)

# get file name
sam_file=ARGV[0]

if ARGV.size<1
	puts "you need a SAM file as input"
	exit
end

# ____________________________for standard deviation calculation

module Enumerable

	def sum
		self.inject(0){|accum, i| accum + i }
	end

	def mean
		self.sum/self.length.to_f
	end

	def sample_variance
		m = self.mean
		sum = self.inject(0){|accum, i| accum +(i-m)**2 }
		sum/(self.length - 1).to_f
	end

	def standard_deviation
	return Math.sqrt(self.sample_variance)
	end
end

array_ins_length=[]

# ______________________________parse SAM file

File.open(sam_file).each do |line|

	array=line.split("\t")

		if (line !~ /^@/)

		seq_name=array[0]
		ins_length=array[8]

			# take insert size for only one sequence
			if (seq_name=~/(\S+)\/([1])/)
```

```
53                              if ins_length.to_i != 0
54                              array_ins_length << ins_length.to_i.abs
55                              end
56                          end
57                  end
58          end
59
60  sum = 0
61  array_ins_length.each { |a| sum+=a }
62
63  # ____________________parameters calculation
64
65  mean=(sum/array_ins_length.length).round(2)
66  sd=array_ins_length.standard_deviation
67
68  # ____________________print insert size parameters
69
70  puts "insert size parameters:"
71  puts "mean: #{mean}"
72  puts "standad_deviation: #[77]"
```

Apendice C: Script para calcular estadísticas sobre marcadores SSR a partir del fichero de salida MREPS

```
#!/usr/bin/env ruby
# Hicham benzekri
# 23-09-2013
# Script para extraer las estadísticas de los marcadores SSRs a partir del fichero
# de salida del programa MREPS

#_____________________________________Input files

file_ssrs=ARGV[0]
unigenes_ids=ARGV[1]
fln_annot=ARGV[2]

if ARGV.length!=3
    puts "Usage:\n\nssrs_stats.rb <file_ssrs> <unigenes_ids> <fln_pt_seqs>"
    puts "or"
    puts "ssrs_stats.rb <file_ssrs> all <fln_pt_seqs>"
    exit
end

#_____________________________________Output files
outfile=File.new("ssrs.txt","w")
ung_with_ssrs=File.new("ung_with_ssrs.txt","w")
ssrs_table=File.new("ssrs_table.txt","w")

#_____________________________________Variables
hash_unigenes={}
hash_utr_pos={}
hash_motifs={}
hash_most_ab_motif={}
array=[]
array_tmp=[]
unigene_name=""
sum_ssrs=0

motif2=0
motif2_utr=0
motif2_orf=0
motif2_unknown=0

motif3=0
motif3_utr=0
motif3_orf=0
motif3_unknown=0

motif4=0
motif4_utr=0
motif4_orf=0
motif4_unknown=0

motif_others=0
motif_others_utr=0
motif_others_orf=0
```

```
53  motif_others_unknown=0
54
55  motif_gata=0
56  motif_gata_utr=0
57  motif_gata_orf=0
58  motif_gata_unknown=0
59
60  #_____________________________________Parsing MREPS file
61
62  File.open(file_ssrs).each do |line|
63
64      line.chomp!
65      if line=~/Processing\s+sequence\s+[']([\S+)[']/
66          unigene_name=$1
67          if(hash_unigenes[unigene_name]==nil)
68              hash_unigenes[unigene_name]=[]
69          end
70      end
71
72      if line=~/^\s+(\d+)\D+(\d+)\D+(\d+)[^[:alpha:]]+([\w\s]+)/
73
74          array_tmp=[]
75          array_tmp << $1.to_i
76          array_tmp << $2.to_i
77          array_tmp << $4.chomp
78
79           sequence=$4.chomp
80
81          if(sequence=~/(\w+)\s.+/)
82              motif=$1
83          end
84          array_tmp << motif
85          hash_unigenes[unigene_name] << array_tmp
86      end
87  end
88
89  #_____________________________________Parsing unigenes ids file
90  hash_ids={}
91  if (unigenes_ids!="all")
92      File.open(unigenes_ids).each do |line|
93          ung=line.chomp
94          hash_ids[ung]=true
95      end
96  end
97
98  #_____________________________________Parsing Full-LengtherNext pt_seqs file
99
100 File.open(fln_annot).each do |line|
101     line.chomp!
102     array=line.split("\t")
103     unigene_name=array[0]
104     orf_start=array[11].to_i
105     orf_end=array[12].to_i
106     hash_utr_pos[unigene_name]=[orf_start,orf_end]
107 end
108
109 #_____________________________________Statistics calculation
```


```
110
111 hash_unigenes.each do |ung,array|
112
113     if (unigenes_ids=="all")
114         hash_ids[ung]=true
115     end
116
117     if (hash_ids[ung]==true)
118
119     sum_ssrs+=hash_unigenes[ung].length
120
121         if (hash_unigenes[ung].length!=0)
122             ung_with_ssrs.puts ung
123         end
124
125         array.each do |a|
126
127             ssr_start=a[0]
128             ssr_end=a[1]
129             ssr_seq=a[2]
130             ssr_motif=a[42]
131
132             orf_start=0
133             orf_end=0
134
135             #_____________________________________________
136
137             orig_sequence=ssr_seq
138             sequence=orig_sequence.upcase
139             array_motif=sequence.split(" ")
140             #_____________________________________________
141
142             if hash_most_ab_motif[array_motif.first]==nil
143                 hash_most_ab_motif[array_motif.first]=1
144             else
145                 hash_most_ab_motif[array_motif.first]+=1
146             end
147             #_____________________________________________
148
149             if array_motif.first.length!=array_motif.last.length
150                 array_motif.pop
151             end
152
153             if hash_motifs["#{array_motif.first} #{array_motif.length}"]==nil
154                 hash_motifs["#{array_motif.first} #{array_motif.length}"]=1
155             else
156                 hash_motifs["#{array_motif.first} #{array_motif.length}"]+=1
157             end
158             #_____________________________________________
159
160             if(hash_utr_pos[ung]!=nil)
161                 orf_start=hash_utr_pos[ung][0]
162                 orf_end=hash_utr_pos[ung][1]
163             else
164                 orf_start="none"
165                 orf_end="none"
166             end
```

```
167             #_____________________________________________
168             if (ssr_motif.length==2)
169                 motif2+=1
170                 if (orf_start.to_s!="none")
171                     if (ssr_start>=orf_start and ssr_end<=orf_end)
172                         motif2_orf+=1
173                     else
174                         motif2_utr+=1
175                     end
176                 else
177                     motif2_unknown+=1
178                 end
179             end
180
181             if (ssr_motif.length==3)
182                 motif3+=1
183                 if (orf_start.to_s!="none")
184
185                     if (ssr_start>=orf_start and ssr_end<=orf_end)
186                         motif3_orf+=1
187                     else
188                         motif3_utr+=1
189                     end
190                 else
191                     motif3_unknown+=1
192                 end
193             end
194
195             if (ssr_motif.length==4)
196                     motif4+=1
197                 if (orf_start.to_s!="none")
198                     if (ssr_start>=orf_start and ssr_end<=orf_end)
199                     motif4_orf+=1
200                     else
201                     motif4_utr+=1
202                     end
203                 else
204                     motif4_unknown+=1
205                 end
206             end
207
208             if (ssr_motif.length>4)
209             motif_others+=1
210                 if (orf_start.to_s!="none")
211                     if (ssr_start>=orf_start and ssr_end<=orf_end)
212                     motif_others_orf+=1
213                     else
214                     motif_others_utr+=1
215                     end
216                 else
217                     motif_others_unknown+=1
218                 end
219             end
220
221             if (ssr_motif=="GATA" or ssr_motif=="TATC")
222                 motif_gata+=1
223                 if (orf_start!="none")
```

```
                    if (ssr_start>=orf_start and ssr_end<=orf_end)
                    motif_gata_orf+=1
                    else
                    motif_gata_utr+=1
                    end
                else
                    motif_gata_unknown+=1
                end
            end

            #_____________________________________________

            outfile.puts "#{ung}\t#{a.join("\t")}"
        end

        end

    end

hash_motifs_sd_phase={}

hash_motifs.sort{|a,b| b[1]<=>a[1]}.each do |motif_rep,count|
motif=motif_rep.split(" ").first.chomp
rep=motif_rep.split(" ").last.chomp.to_i


if (hash_motifs_sd_phase[motif]==nil)
    hash_motifs_sd_phase[motif]=[]
    end
    hash_motifs_sd_phase[motif] << [rep.to_i,count.to_i]
end


maximum=0
hash_motifs_sd_phase.sort{|a,b| a[0].length<=>b[0].length}.each do |motif,array|
    array.each do |sub_array|
        if sub_array.first>maximum
            maximum=sub_array.first
        end

    end
end

#_____________________________________Print statistics table

ssrs_table.print "\t"

(1..maximum).each do |i|
    ssrs_table.print "#{i}\t"
end
ssrs_table.print "Total\n"

hash_motifs_sd_phase.sort{|a,b| a[0].length<=>b[0].length}.each do |motif,array|

    ssrs_table.print "#{motif}\t"
    total=0
    (1..maximum).each do |i|
```

```
281             p=false
282             array.each do |sub_array|
283                 if sub_array.first==i
284                     ssrs_table.print "#{sub_array.last}\t"
285                     total+=sub_array.last
286                     p=true
287                 end
288             end
289
290             if p==false
291                 ssrs_table.print "\t"
292             end
293
294         end
295
296         ssrs_table.print total
297         ssrs_table.print "\n"
298     end
299
300     max_count=0
301     max_motif=""
302
303     hash_most_ab_motif.each do |motif,count|
304         if count>max_count
305             max_count=count
306             max_motif=motif
307         end
308     end
309
310     #___________________________________Print SSRs Summary
311
312     puts "Total SSRs: #{ sum_ssrs}",""
313
314     puts "di-nucleotide:\t#{motif2}"
315     puts "di-nucleotide_orf:\t#{motif2_orf}"
316     puts "di-nucleotide_utr:\t#{motif2_utr}"
317     puts "di-nucleotide_unk:\t#{motif2_unknown}",""
318
319     puts "tri-nucleotide:\t#{motif3}"
320     puts "tri-nucleotide_orf:\t#{motif3_orf}"
321     puts "tri-nucleotide_utr:\t#{motif3_utr}"
322     puts "tri-nucleotide_unk:\t#{motif3_unknown}",""
323
324     puts "tetra-nuleotide:\t#{motif4}"
325     puts "tetra-nuleotide_orf:\t#{motif4_orf}"
326     puts "tetra-nuleotide_utr:\t#{motif4_utr}"
327     puts "tetra-nuleotide_unk:\t#{motif4_unknown}",""
328
329     puts "others:\t#{motif_others}"
330     puts "others_orf:\t#{motif_others_orf}"
331     puts "others_utr:\t#{motif_others_utr}"
332     puts "others_unk:\t#{motif_others_unknown}",""
333
334     puts "motif_gata:\t#{motif_gata}"
335     puts "motif_gata_orf:\t#{motif_gata_orf}"
336     puts "motif_gata_utr:\t#{motif_gata_utr}"
337     puts "motif_gata_unk:\t#{motif_gata_unknown}",""
```

```
338
339   puts "most_abundant_motif: #{max_motif} (#{max_count})"
340   #_____________________________________________
341
342   ung_with_ssrs.close
343   outfile.close
344   ssrs_table.close
```

Apendice D: Software utilizado en las diferentes etapas del análisis de los transcriptomas y genomas estudiados

Estudio	Etapa del analisis	Software utilizado
Transcriptoma de *Solea senegalensis* (v4)	Preprocesamiento	SeqTrimNext
	Ensamblaje y testeo de calidad	MIRA EULER OASES Full-LengtherNext BOWTIE 2 CD-HIT CAP3
	Annotación	Full-LengtherNext Sma3s AutoFact Mpileup (Samtools) MREPS
Transcriptoma de *Solea solea* (v1)	Preprocesamiento	SeqTrimNext
	Ensamblaje	OASES CD-HIT CAP3
	Annotación	Full-LengtherNext Sma3s AutoFact Mpileup (Samtools) MREPS
Transcriptoma de *Tisochrysis lutea* (v2)	Preprocesamiento	SeqTrimNext
	Ensamblaje y testeo de calidad	MIRA EULER Full-LengtherNext BOWTIE 2 SOAPdenovo CAP3
	Annotación	Full-LengtherNext Sma3s AutoFact Mpileup (Samtools) MREPS

Apendice D: Continuación

Estudio	Etapa del analisis	Software utilizado
Transcriptoma de *Ruditapes decussatus* (v1)	Preprocesamiento	SeqTrimNext
	Ensamblaje	SOAPdenovo CD-HIT
	Annotación	Full-LengtherNext Sma3s AutoFact Mpileup (Samtools) MREPS
Genoma de *Photobacterium damselae*	Preprocesamiento	SeqTrimNext
	Ensamblaje	CABOG SOAPdenoo Gap Closer SSPACE
	Annotación	RAST
	Estudio comparativo	PROSPLIGN SEED Viewer
	Busqueda de plasmidos	BOWTIE 2
Genoma de *Solea senegalensis*	Preprocesamiento	SeqTrimNext
	Ensamblaje y testeo de calidad	RAY GAM-NGS Nucmer ICMapper SOAPdenovo Scaffolder SOAPdenovo Gap Closer SSPACE Scaffolder
	Estudio comparativo (con *Cynoglossus semilaevis*)	BLAST SPLIGN PROSPLIGN GEvo ICMapper